PRIVACY-PRESERVING STATISTICAL TOOLS: DIFFERENTIAL PRIVACY AND BEYOND

A Dissertation
Presented to
The Academic Faculty

By

Wanrong Zhang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Engineering
H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology

May 2021

PRIVACY-PRESERVING STATISTICAL TOOLS: DIFFERENTIAL PRIVACY AND BEYOND

Thesis committee:

Dr. Rachel Cummings, advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Sara Krehbiel
Department of Mathematics and Computer Science
Santa Clara University

Dr. Yajun Mei, advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Jeff Wu
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Mark Davenport
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date approved: April 21, 2021

To my beloved mom and grandparents

ACKNOWLEDGMENTS

I would first like to express my sincere gratitude to my advisors, Rachel Cummings and Yajun Mei. I was very fortunate to have them sharing knowledge in both statistics and computer science. Yajun is a great statistician and teacher. It was him who first guided me as a fresh Ph.D. student to the research world. He not only taught me knowledge in mathematical statistics, but also supported me to explore several research directions and find my research interest. I am so grateful to also have Rachel as my advisor since my second year when she joined Georgia Tech. She introduced me to the exciting research area in privacy and security, and has always been supportive in every angle of my research. I wish to highlight her contagious enthusiasm for research, which encouraged me to pursue a career in research. This thesis would never be possible without my two advisors' patient support, valuable guidance, and all the opportunities to further my research. They helped me to sharpen my thinking and become an independent researcher.

I would also like to express appreciation to my thesis committee members: Sara Krehbiel, for her sage advice and being an incredible collaborator; Jeff Wu, for his insightful feedback and discussion on my thesis proposal; and Mark Davenport, for providing valuable input on my thesis.

I visited several wonderful institutions during my Ph.D.: the Simons Institute for the Theory of Computing, Microsoft Research Cambridge, and Microsoft Research Redmond. I owe my gratitude to all my hosts and friends, who helped shed light on many of my ideas, including but not limited to Jana Kulkarni, Robert Sim, Nalin Singal and Priyanka Kulkarni. I would also like to thank all my collaborators include: Olya Ohrimenko, for her patient mentorship and being a constant source of inspiration; Gautam Kamath, for fruitful discussions and encouragement and his tweets to increase my visibility in my research community; Rui Tuo, for introducing me many useful advanced probability tools that have a profound impact on my research; Shruti Tople, for sharing her expertise on security attacks;

Yuliia Lut, for being a reliable collaborator and generous friend. All of their expertise has been greatly beneficial to my research and career development.

I was very fortunate to meet many amazing friends at Georgia Tech, Simons, and Microsoft Research, including but not limited to Shanshan, Liyan, Junqi, Chelsea, Chuanping, Siawpeng, Brendan, and Jack. They made my experience in graduate school so enjoyable. Especially, I want to thank Helen and Ruqi, who have always been there for me. Thanks for the calls, tea times, and all the fun and moral support. I also want to thank Juba for his help on my job search, and Nancy for helping me settle down in Atlanta. Special thanks to my dear hamster for the companionship and comfort during the five years, and especially the quarantine time.

Finally, but not least, I have my most heartfelt thanks to my mom and my grandparents for their love and support throughout my life. Without them, I would never have enjoyed so many opportunities and experiences that have made me who I am. My cousin deserve my wholehearted thanks as well.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Differential privacy has emerged as the de facto gold standard in protecting the privacy of individuals when processing sensitive data, because of its powerful formal guarantees. Several companies, including Google, Apple, Microsoft, and the U.S. Census Bureau, have deployed differentially private tools, but barriers remain between such systems and full-featured privacy-preserving data analytics.

In this thesis, we will focus on two main challenges: private online decision-making and privacy of dataset-level properties. Most of the existing differentially private tools are for static databases with non-adaptive analysis. However, modern data analysts interact with the dataset in an inherently adaptive fashion, and we are confronted with databases that arrive online. To address these challenges, we study private algorithms for two classical statistical online decision-making problems: change-point detection and online false discovery rate control. Second, we demonstrate a new dataset-level privacy vulnerability. Classical differential privacy provides individual-level guarantees, but does not protect sensitive global properties of a dataset, such as the distribution of race and gender among users in the training set or proprietary information. Therefore, we introduce new privacy notions that protect dataset-level attributes and mechanisms that provide formal guarantees beyond individual privacy.

CHAPTER 1
INTRODUCTION AND BACKGROUND

1.1 Introduction

Data-driven algorithmic decision-making plays a central role in critical human-centered applications, determining everything from social media content to job/insurance eligibility to government policy-making. In an ideal world, we might hope that people would be willing to share their data in exchange for convenience or information. However, in reality, in the wake of numerous digital privacy scandals—such as Facebook-Cambridge Analytica data scandal [1]—a majority of Americans are concerned about how both companies and governments collect, store, and use their data [2], and share less information as a result. Increasingly, privacy is not merely a question of philosophy, but table stakes in the course of business. Classical data analyses tools were not developed with privacy in mind and have been shown can unintentionally store and memorize personal data [3, 4, 5, 6].

One might be thinking we can simply anonymize the data—removing personally identifying information in the dataset. The issue is that the information contained in the dataset might be available in other forms on the internet. The richness of the data enables naming an individual by a combination of the attributes, such as a combination of location, gender and income, which allows us to match anonymized records with non-anonymized records available in a different dataset. As an example, Netflix released a customer rating dataset for competition in 2009, and the data was fully anonymized prior to the launch of the competition. However, researchers successfully recovered up to 99% of the identity by using some information publicly available on the Internet Movie Database (IMDb), see [7]. Thus, data cannot be fully anonymized and remain useful.

The field of *differential privacy* [8] overcomes such challenges and offers data analysis

tools that provide powerful worst-case privacy guarantees; it has become a de facto gold standard in privacy-preserving data analysis. Informally, a differentially private algorithm ensures that the output of the algorithm will still have approximately the same distribution when a small number of data entries are changed. The privacy is quantified by a parameter ϵ. This parameterized privacy notion allows for a smooth tradeoff between accurate analysis and privacy to the individuals. Recently, we have seen several real-world deployments at Google [9], Apple [10], Microsoft [11], and the U.S. Census Bureau [12]. yet, there are still barriers to fully deploy differential privacy in practice. First, most of the existing tools are for static databases with non-adaptive analysis, while modern data analysts interact with the data set in an inherently adaptive fashion, and we are confronted with databases that arrive online. To address this challenge, in this thesis, we study two private statistical online/sequential decision-making problems, including change-point detection and online false discovery rate control—both are fundamental and widely used in data analytics. Second, differential privacy does not protect sensitive global properties of a dataset, such as the distribution of race and gender among users in the training set or proprietary information. In this thesis, We identify the dataset-level privacy vulnerabilities in collaborative learning and propose the appropriate privacy notions to protect the vulnerabilities beyond individual privacy.

1.1.1 Private Online Statistical Decision-making

In Chapter 2, based on joint work with Rachel Cummings, Sara Krehbiel, Rui Tuo, and Yajun Mei that appeared in NeurIPS 2018 [13] and Journal of Machine Learning Research in 2021 [14], we study the statistical problem of change-point detection through the lens of differential privacy. The change-point detection problem seeks to identify distributional changes at an unknown change-point k^* in a stream of data. Increasingly, tools for change-point detection are applied in settings where data may be highly sensitive and formal privacy guarantees are required, such as identifying disease outbreaks based on

hospital records, or IoT devices detecting activity within a home. We give the first private algorithms for both online and offline change-point detection. We prove a differential privacy guarantee for our algorithms and accuracy guarantees that bound the additive error of our estimate of the true change-point with high probability. Since traditional statistics typically focuses on the asymptotic consistency and unbiasedness of the estimator, we also give the first finite-sample accuracy guarantees for the standard (non-private MLE). Additionally, we provide empirical validation, which enhances our theoretical results, and provides evidence that our algorithms perform well for practical use.

In Chapter 3, based on joint work with Rachel Cummings and Gautam Kamath [15], we study False Discovery Rate (FDR) control in multiple hypothesis testing under the constraint of differential privacy for the sample. In hypothesis testing, a false discovery occurs when a hypothesis is incorrectly rejected due to noise in the sample. When adaptively testing multiple hypotheses, the probability of a false discovery increases as more tests are performed. Thus the problem of False Discovery Rate (FDR) control is to find a procedure for testing multiple hypotheses that accounts for this effect in determining the set of hypotheses to reject. The goal is to minimize the number (or fraction) of false discoveries, while maintaining a high true positive rate (i.e., correct discoveries). Unlike previous work in this direction, we focus on the online setting, meaning that a decision about each hypothesis must be made immediately after the test is performed, rather than waiting for the output of all tests as in the offline setting. We provide new private algorithms based on state-of-the-art results in non-private online FDR control. Our algorithms have strong provable guarantees for privacy and statistical performance as measured by FDR and power. We also provide experimental results to demonstrate the efficacy of our algorithms in a variety of data environments.

1.1.2 Beyond Differential Privacy

Modern machine learning models have been shown to memorize information about their training data, leading to privacy concerns regarding their use and release in practice [4, 5]. Much attention has been devoted to identifying and preventing vulnerabilities of individual privacy. However, in many settings, the global properties of a dataset may also be sensitive—e.g., the distribution of race and gender among user in the dataset for training a voice or facial recognition models, the overall mortality rate in a hospital, and proprietary information such as sales distribution of products.

In Chapter 4, based on joint work with Shruti Tople and Olya Ohrimenko [16], we study the leakage of dataset properties at the population-level. Concerns of leaking global properties of a sensitive attribute (e.g., sales numbers or gender) through training machine-learned models arise in several different communication models, including a single party releasing a trained model and in multi-party machine learning. Our primary focus in this work is on attacks to infer dataset properties in the centralized multi-party machine learning setting, where the model is securely trained on several parties' data, and parties only have black-box access to the final model. We propose an effective attack strategy that requires only a few hundred queries to the model and relies on a simple attack architecture that even a computationally bound attacker can use. We show that global properties about one of the parties' sensitive attributes can be inferred by the second party, even when the attacker has limited access to the model. Our attack successes on different types of datasets including tabular, text, and graph data. To understand and measure the source of leakage, we consider several models of correlation between a sensitive attribute and the rest of the data. Using multiple machine learning models, we show that leakage occurs even if the sensitive attribute is not included in the training data and has a low correlation with other attributes and the target variable. Our attack suggests that sensitive attribute leakage is a significant privacy vulnerability, motivating our work in chapter 5 that studies methods for protecting dataset-level attribute privacy.

Differential privacy guarantees individual-level privacy when publishing an output computed on a database, but it does not aim to protect population-level information, and was designed to learn the global properties of a dataset without sacrificing individual privacy. The study of tools to protect *attribute privacy*, where an analyst must prevent global properties of sensitive attributes in her dataset from leaking during analysis, is limited, both in terms of a framework for reasoning about it and mechanisms for protecting it. In Chapter 5, based on joint work with Rachel Cummings and Olya Ohrimenko [17], we depart from individual privacy to initiate the study of attribute privacy, where a data owner is concerned about revealing sensitive properties of a whole dataset during analysis. We propose definitions to capture *attribute privacy* in two relevant cases where global attributes may need to be protected:(1) properties of a specific dataset and (2) parameters of the underlying distribution from which dataset is sampled. We also provide two efficient mechanisms and one inefficient mechanism that satisfy attribute privacy for these settings. We base our results on a novel use of the Pufferfish framework to account for correlations across attributes in the data, thus addressing" the challenging problem of developing Pufferfish instantiations and algorithms for general aggregate secrets" that was left open by[18].

1.2 Background on Differential Privacy

This section introduces the relevant background on differential privacy that will be used throughout this thesis. We begin in Section 1.2.1 with the formal definition of differential privacy and several properties. Section 1.2.2 provides a brief introduction to several fundamental algorithmic tools for differential privacy, which can be used as subroutines to design more sophisticated algorithms. For details, we refer the interested reader to [19].

1.2.1 Definition and Properties

Differential privacy bounds the maximum amount that a single data entry can affect analysis performed on the database. Two databases X, X' are *neighboring* if they differ in at most

one entry.

Definition 1 (Differential Privacy [8])**.** *An algorithm $\mathcal{M} : \mathbb{R}^n \to \mathcal{R}$ is* (ϵ, δ)-differentially private *if for every pair of neighboring databases $X, X' \in \mathbb{R}^n$, and for every subset of possible outputs $\mathcal{S} \subseteq \mathcal{R}$,*

$$\Pr[\mathcal{M}(X) \in \mathcal{S}] \leq \exp(\epsilon) \Pr[\mathcal{M}(X') \in \mathcal{S}] + \delta.$$

If $\delta = 0$, we say that $\mathcal{M}$ is ϵ-differentially private.

Differential privacy has a number of desirable properties. First, differential privacy is robust to *post-processing*. It means no data analysts can increase privacy loss by taking a differentially private output and performing additional computations.

Theorem 1. *Let $\mathcal{M} : \mathbb{R}^n \to \mathcal{R}$ be a randomized algorithm that is (ϵ, δ)-differentially private. Let $f : \mathcal{R} \to \mathcal{R}'$ be an arbitrary randomized mapping. Then $f \cdot \mathcal{M} : \mathbb{R}^n \to \mathcal{R}'$ is (ϵ, δ)-differentially private.*

Differentially private algorithms compose. It means the privacy loss degrades gracefully over multiple computations, which allows us to design differentially private algorithms from several building blocks.

Theorem 2. *Let $\mathcal{M}_i : \mathbb{R}^n \to \mathcal{R}_i$ be an $(\epsilon_i, \delta_i)-$differentially private algorithm for $i \in [1, 2, \ldots, k]$. Then if $\mathcal{M}_{[k]} : \mathbb{R}^n \to \Pi_{i=1}^k \mathcal{R}_i$ is defined to be $\mathcal{M}_{[k]}(x) = (\mathcal{M}_1(x), \ldots, \mathcal{M}_k(x))$, then $\mathcal{M}_{[k]}$ is $(\sum_{i=1}^k \epsilon_i, \sum_{i=1}^k \delta_i)$-differentially private.*

Differential privacy allows the analysis and control of privacy loss incurred by groups, with the strength of the privacy guarantee drops linearly with the size of the group. Any ϵ-differentially private mechanism $\mathcal{M}$ is $k\epsilon$-differentially private for groups of size k.

1.2.2 Algorithmic Tools for Differential Privacy

The *additive sensitivity* of a real-valued query $f : \mathcal{X}^n \to \mathbb{R}$ is denoted Δf, and is defined to be the maximum change in the function's value that can be caused by changing a single entry. That is,

$$\Delta f = \max_{X, X' \text{ neighbors}} |f(X) - f(X')| .$$

If f is a vector-valued query, the expression above can be modified with the appropriate norm in place of the absolute value. Differential privacy guarantees are often achieved by adding *Laplace noise* at various places in the computation, where the noise scales with $\Delta f/\epsilon$, where Δf is the additive sensitivity with ℓ_1-norm. A Laplace random variable with parameter b is denoted $\text{Lap}(b)$, and has probability density function,

$$p_{\text{Lap}(b)}(x) = \frac{1}{2b} \exp\left(\frac{-|x|}{b}\right) \quad \forall x \in \mathbb{R}.$$

We may sometimes abuse notation and also use $\text{Lap}(b)$ to denote the realization of a random variable with this distribution.

An alternative to adding Laplacian noise is to add Gaussian noise. In this case, rather than scaling the noise to the ℓ_1-sensitivity, we instead scale to the ℓ_2 sensitivity. The Gaussian mechanism with parameter b adds zero-mean Gaussian noise with variance b in each of the k coordinates.

Theorem 3 ([19]). *The Gaussian mechanism with parameter $\sigma \geq c\Delta_2(f)/\epsilon$ is $(\epsilon, \delta)-$differentially private for $c^2 > 2\log(1.25/\delta)$.*

Differentially private selection is one of the fundamental problems: the space of outcomes is discrete and the task is to produce a "best" answer. A common algorithm to determine which of m queries with sensitivity Δ has the highest value is Report Noisy Max [20, 19]. It Adds independently generated Laplace noise $\text{Lap}(\Delta/\epsilon$ to each count and returns the index of the largest noisy count (we ignore the possibility of a tie).

Algorithm 1 Report Noisy Max: REPORTMAX($X, \Delta, \{f_1, \ldots, f_m\}, \epsilon$)

Input: database X, set of queries $\{f_1, \ldots, f_m\}$ each with sensitivity Δ, privacy parameter ϵ

for $i = 1, \ldots, m$ **do**

Compute $f_i(X)$

Sample $Z_i \sim \text{Lap}(\frac{\Delta}{\epsilon})$

end for

Output $i^* = \underset{i \in [m]}{\text{argmax}} \left(f_i(X) + Z_i\right)$

Theorem 4 ([20, 19])**.** REPORTMAX *is ϵ-differentially private.*

The exponential mechanism [19] is another common algorithm for differentially private selection, and it was designed for situations in which we wish to choose the "best" response to queries with arbitrary utilities while preserving differential privacy. Given some arbitrary output range $\mathcal{R}$, the exponential mechanism is defined with respect to some utility function u, which maps databases and output pairs to utility scores. The sensitivity of the utility function is thus $\Delta u = \max_{r \in \mathcal{R}} \max_{x,y:||x-y||_1 \leq 1} |u(x,r) - u(y,r)|$. The exponential mechanism outputs an element $r \in \mathcal{R}$ with probability proportional to $\exp(\epsilon u(x,r)/(2\Delta u))$.

Theorem 5 ([19])**.** *The exponential mechanism is ϵ-differentially private.*

In some situations, we might be facing a very large number of questions to answer, but we only care to know the answers of the queries that lie above a certain threshold. In the offline case, this problem can be handled by running the REPORTMAXor the exponential mechanism iteratively. In the online setting, we introduce the ABOVETHRESH and SPARSEVECTOR techniques.

The ABOVETHRESH algorithm, first introduced by [21] and refined to its current form by [19], takes in a potentially unbounded stream of queries, compares the answer of each

query to a fixed noisy threshold, and halts when it finds a noisy answer that exceeds the noisy threshold. We state the privacy and accuracy guarantees of ABOVETHRESH below.

Algorithm 2 Above Noisy Threshold: ABOVETHRESH$(X, \Delta, \{f_1, f_2, \ldots\}, T, \epsilon)$

Input: database X, stream of queries $\{f_1, f_2, \ldots\}$ each with sensitivity Δ, threshold T, privacy parameter ϵ

Let $\hat{T} = T + \text{Lap}(\frac{2\Delta}{\epsilon})$

for each query i **do**

  Let $Z_i \sim \text{Lap}(\frac{4\Delta}{\epsilon})$

  if $f_i(X) + Z_i > \hat{T}$ **then**

    Output $a_i = \top$

    Halt

  else

    Output $a_i = \perp$

  end if

end for

Theorem 6 ([21])**.** ABOVETHRESH *is ϵ-differentially private.*

Theorem 7 ([21])**.** *For any sequence of m queries $f_1, \ldots, f_m$ with sensitivity Δ such that $|\{i < m : f_i(X) \geq T - \alpha\}| = 0$,* ABOVETHRESH *outputs with probability at least $1 - \beta$ a stream of $a_1, \ldots, a_m \in \{\top, \perp\}$ such that $a_i = \perp$ for every $i \in [m]$ with $f(i) < T - \alpha$ and $a_i = \top$ for every $i \in [m]$ with $f(i) > T + \alpha$ as long as*

$$\alpha \geq \frac{8\Delta \log(2m/\beta)}{\epsilon}.$$

The SPARSEVECTOR algorithm can be thought of as making repeated calls to ABOVETHRESH : Each time an above threshold query is reported, the algorithm simply restarts the remaining stream of queries on a new instantiation of ABOVETHRESH . It halts after it has restarted ABOVETHRESH c times. The privacy guarantee follows from the composition theorem. The accuracy guarantee is stated in Theorem 9.

Algorithm 3 Sparse Vector: SPARSEVECTOR($X, \Delta, \{f_1, f_2, \ldots\}, T, c, \epsilon$)

Input: database X, stream of queries $\{f_1, f_2, \ldots\}$ each with sensitivity Δ, threshold T, a cutoff point c, privacy parameter ϵ

Let $\hat{T}_0 = T + \text{Lap}(\frac{2\Delta c}{\epsilon})$

Let count $= 0$

for each query i **do**

  Let $Z_i \sim \text{Lap}(\frac{4\Delta c}{\epsilon})$

  if $f_i(X) + Z_i > \hat{T}$ **then**

    Output $a_i = \top$

    Let count $=$ count $+1$

    Let $\hat{T}_{\text{count}} = T + \text{Lap}(\frac{2\Delta c}{\epsilon})$

  else

    Output $a_i = \perp$

  end if

  if count $\geq c$ **then**

    Halt.

  end if

end for

Theorem 8 ([21])**.** SPARSEVECTOR *is* $(\epsilon, 0)$*-differentially private.*

Theorem 9 ([21])**.** *For any sequence of* k *queries* $f_1, \ldots, f_k$ *with sensitivity* Δ *such that* $|\{i : f_i(X) \geq T - \alpha\}| \leq c$*,* SPARSEVECTOR *outputs with probability at least* $1 - \beta$ *a stream of* $a_1, \ldots, a_k \in \{\top, \perp\}$ *such that* $a_i = \perp$ *for every* $i \in [m]$ *with* $f(i) < T - \alpha_{SV}$ *and* $a_i = \top$ *for every* $i \in [m]$ *with* $f(i) > T + \alpha_{SV}$ *as long as* $\alpha_{SV} \geq \frac{8\Delta c \log(2kc/\beta)}{\epsilon}$*.*

CHAPTER 2

DIFFERENTIALLY PRIVATE CHANGE-POINT DETECTION

2.1 Introduction

The *change-point detection problem* seeks to identify distributional changes at an unknown change-point k^* in a stream of data. The estimated change-point should be consistent with the hypothesis that the data are initially drawn from pre-change distribution P_0 but from post-change distribution P_1 starting at the change-point. This problem appears in many important practical settings, including biosurveillance [22], fault detection [23], finance [24], signal detection [25], and security systems [26, 27]. For example, the CDC may wish to detect a disease outbreak based on real-time data about hospital visits, or smart home IoT devices may want to detect changes in activity within the home. In both of these applications, the data contain sensitive personal information.

In this work we study the statistical problem of change-point detection through the lens of differential privacy. We give private algorithms for both online and offline change-point detection, analyze these algorithms theoretically, and then provide empirical validation of these results.

2.1.1 Related Work

The change-point detection problem originally arose from industrial quality control, and has since been applied in a wide variety of other contexts including climatology [28], econometrics [29], and DNA analysis [30]. The problem is studied both in the *offline setting*, in which the algorithm has access to the full dataset $X = \{x_1, \ldots, x_n\}$ up front, and in the *online setting*, in which data points arrive one at a time $X = \{x_1, \ldots\}$. Change-point detection is a canonical problem in statistics that has been studied for nearly a century;

selected results include [31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45].

Our approach is inspired by the commonly used Cumulative Sum (CUSUM) procedure [32]. It follows the generalized log-likelihood ratio principle, calculating

$$\ell(k) = \sum_{i=k}^{n} \log \frac{P_1(x_i)}{P_0(x_i)}$$

for each $k \in [n]$ and declaring that a change occurs if and only if $\ell(\hat{k}) \geq T$ for MLE $\hat{k} = \operatorname{argmax}_k \ell(k)$ and appropriate threshold $T > 0$. The existing change-point literature works primarily in the asymptotic setting when $k_n^*/n \to r$ for some $r \in (0, 1)$ as $n \to \infty$ (see, e.g., [46, 47]). In contrast, we consider finite databases and provide the first accuracy guarantees for the MLE from a finite sample ($n < \infty$).

In offering the first algorithms for *private* change-point detection, we primarily use two powerful tools from the differential privacy literature. REPORTMAX [20, 19] calculates noisy approximations of a stream of queries on the database and reports which query produced the largest noisy value. We instantiate this with partial log-likelihood queries to produce a private approximation of the the change-point MLE in the offline setting. ABOVETHRESH [21] calculates noisy approximations on a stream of queries on the database iteratively and aborts as soon as a noisy approximation exceeds a specified threshold. We extend our offline results to the harder online setting, in which a bound on k^* is not known a priori, by using ABOVETHRESH to identify a window of fixed size n in which a change is likely to have occurred so that we can call our offline algorithm on that window to estimate the true change-point.

Recently, [48] also provided a private change-point detection algorithm based on the more general problem of private hypothesis testing. Their algorithm partitions time series data into batches of size equal to the sample complexity of the hypothesis testing problem, and then outputs the batch number most consistent with a change-point. Their bound gives the minimum number of data points needed to distinguish between two distributions

with constant advantage but does not necessarily imply the closest possible approximation of the true change-point. Their accuracy guarantees and ours alike are quantified with respect to distance measures between modified versions of the hypothesized distributions, and comparability of the bounds depends on the specific distributions from which data are drawn.

2.1.2 Our Results

We use existing tools from differential privacy to solve the change-point detection problem in both offline and online settings, neither of which have been studied in the private setting before.

Private offline change-point detection. We develop an offline private change-point detection algorithm OFFLINEPCPD (Algorithm 4) that is accurate under one of two assumptions about the distributions from which data are drawn. As is standard in the privacy literature, we give accuracy guarantees that bound the additive error of our estimate of the true change-point with high probability. Our accuracy theorem statements (Theorems 12 and 14) also provide guarantees for the non-private estimator for comparison. Since traditional statistics typically focuses on the the asymptotic consistency and unbiasedness of the estimator, ours are the first finite-sample accuracy guarantees for the standard (non-private) MLE. As expected, MLE accuracy decreases with the sensitivity of the measured quantity but increases as the pre- and post-change distribution grow apart. Interestingly, it is constant with respect to the size of the database. In providing MLE bounds alongside accuracy guarantees for our private algorithms, we are able to quantify the cost of privacy as roughly $D_{KL}(P_0||P_1)/\epsilon$.

We are able to prove ϵ-differential privacy regardless of how the data are generated by instantiating the general-purpose REPORTMAX algorithm from the privacy literature with our log-likelihood queries (Theorem 11). Noting that when the measured quantity

has unbounded sensitivity, we introduce a clamping function so that the sensitivity is still bounded by a certain threshold. Importantly and in contrast to our accuracy results, the distributional assumption need only apply to the hypothesized distributions from which data are drawn; privacy holds for arbitrary input databases.

Private online change-point detection. In ONLINEPCPD (Algorithm 6), we extend our offline results to the online setting by using the ABOVETHRESH framework to first identify a window in which the change is likely to have happened and then call the offline algorithm to identify a more precise approximation of when it occurred. Standard ϵ-differential privacy under our first distributional assumption follows from composition of the underlying privacy mechanisms (Theorem 15). Accuracy of our online mechanism relies on appropriate selection of the threshold that identifies a window in which a change-point has likely occurred, at which point the error guarantees are inherited from the offline algorithm (Theorem 16).

Empirical validation. Finally, we run several Monte Carlo experiments to validate our theoretical results for both the online and offline settings. We consider data drawn from Bernoulli distributions, which satisfies our first distributional assumption, as well as Gaussian and Gamma distributions, which satisfy our second distributional assumptions. Our offline experiments are summarized in 2.1, which shows that change-point detection is easier when P_0 and P_1 are further apart and harder when the privacy requirement is stronger (ϵ is smaller). Additionally, these experiments enhance our theoretical results, finding that OFFLINEPCPD performs well even when we relax the assumptions required for our theoretical accuracy bounds by running our algorithm on imperfect hypotheses P_0 and P_1 that are closer together than the true distributions from which data are drawn. 2.3 shows that ONLINEPCPD also performs well, consistent with our theoretical guarantees.

2.2 Preliminaries

Our work considers the statistical problem of change-point detection through the lens of differential privacy. Section 2.2.1 defines the change-point detection problem, Section 2.2.2 describes how we apply the differentially private tools, and Section 2.2.3 give several concentration inequalities which will be used in our proofs.

2.2.1 Change-point Background

Let $X = \{x_1, \ldots, x_n\}$ be n real-valued data points. The *change-point detection problem* is parametrized by two distributions, P_0 and P_1. The data points in X are hypothesized to initially be sampled i.i.d. from P_0, but at some unknown change time $k^* \in [n]$, where $[n]$ denotes the set $\{1, \ldots, n\}$, an event may occur (e.g., epidemic disease outbreak) and change the underlying distribution to P_1. The goal of a data analyst is to announce that a change has occurred as quickly as possible after k^*. Since the x_i may be sensitive information—such as individuals' medical information or behaviors inside their home—the analyst will wish to announce the change-point time in a privacy-preserving manner.

In the standard non-private offline change-point literature, the analyst wants to test the null hypothesis $H_0 : k^* = \infty$, where $x_1, \ldots, x_n \sim_{\text{iid}} P_0$, against the composite alternate hypothesis $H_1 : k^* \in [n]$, where $x_1, \ldots, x_{k^*-1} \sim_{\text{iid}} P_0$ and $x_{k^*}, \ldots, x_n \sim_{\text{iid}} P_1$. The log-likelihood ratio of $k^* = \infty$ against $k^* = k$ is given by

$$\ell(k, X) = \sum_{i=k}^{n} \log \frac{P_1(x_i)}{P_0(x_i)}. \tag{2.1}$$

The maximum likelihood estimator (MLE) of the change time k^* is given by

$$\hat{k}(X) = \text{argmax}_{k \in [n]} \ell(k, X). \tag{2.2}$$

When X is clear from context, we will simply write $\ell(k)$ and $\hat{k}$.

We always use $\log$ to refer to the natural logarithm, and when necessary, we interpret $\log \frac{0}{0} = 0$. An important quantity in our accuracy analysis will be the Kullback-Leibler divergence between probability distributions P_0 and P_1, defined as $D_{KL}(P_1||P_0) = \int_{-\infty}^{\infty} P_1(x) \log \frac{P_1(x)}{P_0(x)} dx = \mathbb{E}_{x\sim P_1}[\log \frac{P_1(x)}{P_0(x)}]$. For given distributions P_0, P_1, our proofs will use the following three variations of KL-divergence:

$$C = \min \left\{ D_{KL}(P_0||P_1), D_{KL}(P_1||P_0) \right\}, \tag{2.3}$$

$$C_M = \min \left\{ D_{KL}\left(P_0||\frac{P_0+P_1}{2}\right), D_{KL}\left(P_1||\frac{P_0+P_1}{2}\right) \right\} = \min_{i=0,1} \mathbb{E}_{x\sim P_i}\left[\log \frac{2P_i(x)}{P_0(x)+P_1(x)}\right], \tag{2.4}$$

$$C_A = \min \left\{ -\mathbb{E}_{x\sim P_0}\left[\log \frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2}, \mathbb{E}_{x\sim P_1}\left[\log \frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2} \right\}, \tag{2.5}$$

where A is a pre-specified (input) truncation parameter. The truncation operation is defined as

$$[x]_{-A/2}^{A/2} = \begin{cases} -A/2 & if \quad x < -A/2 \\ A/2 & if \quad x > A/2 \\ x & otherwise. \end{cases}$$

We will measure the additive error of our estimations of the true change point as follows.

Definition 2 ((α, β)-accuracy)**.** *A change-point detection algorithm that produces a change-point estimator $\tilde{k}(X)$ where a distribution change occurred at time k^* is* (α, β)-accurate *if* $\Pr[|\tilde{k} - k^*| < \alpha] \geq 1 - \beta$*, where the probability is taken over the randomness of the algorithm and sampling of X.*

2.2.2 Differential Privacy Tools

The *sensitivity* of a function or query f is defined as $\Delta(f) = \max_{\text{neighbors } X,X'} |f(X) - f(X')|$. Since our algorithms estimate a change-point based on log-likelihood ratios, it will be useful to denote the sensitivity of the log-likelihood function given distributions P_0, P_1 as follows:

$$\Delta(\ell) \quad = \quad \max_{x \in \mathbb{R}} \log \frac{P_1(x)}{P_0(x)} - \min_{x' \in \mathbb{R}} \log \frac{P_1(x')}{P_0(x')}. \tag{2.6}$$

Our algorithms rely on two existing differentially private algorithms, REPORTMAX [20, 19] in Algorithm 1 and ABOVETHRESH [21] in Algorithm 2. We use the REPORTMAX algorithm as the framework for our offline private change-point detector OFFLINEPCPD in Section 2.3 to privately select the time k with the highest log-likelihood ratio $\ell(k)$. We use the ABOVETHRESH algorithm as a framework for our online private change-point detector ONLINEPCPD in Section 2.4 when new data points arrive online in a streaming fashion.

2.2.3 Concentration Inequalities

Our proofs will use the following bounds on partial sums of independent random variables.

Lemma 1 (Ottaviani's inequality [49]). *For independent random variables $U_1, \ldots, U_m$, for $S_k = \sum_{i \in [k]} U_i$ for $k \in [m]$, and for $\lambda_1, \lambda_2 > 0$, we have*

$$\Pr\left[\max_{k \in [m]} |S_k| > \lambda_1 + \lambda_2\right] \leq \frac{\Pr\left[|S_m| > \lambda_1\right]}{1 - \max_{k \in [m]} \Pr\left[|S_m - S_k| > \lambda_2\right]}.$$

If we additionally assume the U_i above are i.i.d. with mean 0 and take values from an interval of bounded length L, we can apply Hoeffding's inequality for the following corollary:

Corollary 2. *For independent and identically distributed random variables $U_1, \ldots, U_m$ with mean zero and support strictly bounded by an interval of length L, for $S_k = \sum_{i \in [k]} U_i$*

for $k \in [m]$, and for $\lambda_1, \lambda_2 > 0$, we have

$$\Pr[\max_{k\in[m]} |S_k| > \lambda_1 + \lambda_2] \leq \frac{2\exp(-2\lambda_1^2/(mL^2))}{1 - 2\exp(-2\lambda_2^2/(mL^2))}.$$

When our random variables do not come from a bounded-length interval, we will require Bernstein's inequality instead of Hoeffding's to attain a similar result on their partial sums.

Lemma 3 (Bernstein's inequality [49])**.** *For independent random variables $Y_1, \ldots, Y_m$ with mean zero such that $\mathbb{E}\left[e^{|Y_i|/M} - 1 - \frac{|Y_i|}{M}\right] M^2 \leq \frac{1}{2} v_i$ for constants M and v_i for all $i \in [m]$, we have*

$$\Pr[|Y_1 + \ldots + Y_n| > x] \leq 2\exp\left(-\frac{1}{2}\frac{x^2}{v + Mx}\right),$$

for $v \geq v_1 + \ldots + v_m$.

Corollary 4. *For independent and identically distributed random variables $Y_1, \ldots, Y_m$ with mean zero such that $\mathbb{E}\left[e^{|Y_i|} - 1 - |Y_i|\right] \leq \frac{1}{2} v$, for constant v for all $i \in [m]$, and for $S_k = \sum_{i\in[k]} Y_i$ for $k \in [m]$, and for $\lambda_1, \lambda_2 > 0$, we have*

$$\Pr[\max_{k\in[m]} |S_k| > \lambda_1 + \lambda_2] \leq \frac{2\exp(-\lambda_1^2/(2mv + 2\lambda_1))}{1 - 2\exp(-\lambda_2^2/(2mv + 2\lambda_2))}.$$

2.3 Offline Private Change-point Detection

In this section, we investigate the differentially private change-point detection problem in the setting that n data points $X = \{x_1, \ldots, x_n\}$ are known to the algorithm in advance. Given two hypothesized distributions P_0 and P_1, our algorithms privately approximate the MLE $\hat{k}$ of the change time k^*. We consider accuracy of change-point estimation with and without the assumption that the distributions have uniformly bounded likelihood ratios.

First, we provide finite-sample accuracy guarantees for the MLE in each of these cases in Section 2.3.1. Second, we offer an algorithm OFFLINEPCPD in Section 2.3.2 that

achieves privacy by introducing noise proportional to the sensitivity of the log-likelihood calculation. To detect changes in certain distributions such as Gaussians, our OFFLINEPCPD algorithm requires infinite noise and therefore provides no accuracy. Therefore, we finally provide a second private algorithm OFFLINEPTCPD in Section 2.3.3, which has no restriction on the distributions and instead uses a truncation parameter $A > 0$ to control the sensitivity of the log-likelihood calculation. In 2.1 we summarize accuracy bounds for both the MLE and the output of our algorithms under these assumptions.

Table 2.1: Summary of accuracy guarantees for non-private and private offline change-point detection under the alternate hypothesis H_1. The expressions $\hat{k}$, $\Delta(\ell)$, C, C_M and C_A are defined in (2.2), (2.6), (2.3), (2.4), (2.5), respectively.

Quantity	Accuracy guarantee α
MLE $\hat{k}$	$\min\left\{\frac{2\Delta(\ell)^2}{C^2}\log\frac{32}{3\beta}, \frac{35}{C_M^2}\log\frac{32}{3\beta}\right\}$
OFFLINEPCPD	$\max\left\{\frac{8\Delta(\ell)^2}{C^2}\log\frac{64}{3\beta}, \frac{8\Delta(l)}{C\epsilon}\log\frac{64\Delta(l)}{\beta C\epsilon}\right\}$
OFFLINEPTCPD	$\max\left\{\frac{8A^2}{C_A^2}\log\frac{64}{3\beta}, \frac{8A}{C_A\epsilon}\log\frac{64A}{\beta C_A\epsilon}\right\}$

Although our algorithms only guarantee accuracy if the analyst supplies the true distributions P_0, P_1 from which data are drawn, it is important to note that the algorithms are ϵ-differentially private for any *hypothesized* distributions P_0, P_1 and privacy parameter $\epsilon > 0$ *regardless of the distributions from which X is drawn*. In the change-point or statistical process control (SPC) literature, when the pre- and post- change distributions are unknown in practical settings, researchers often choose hypotheses P_0, P_1 with the smallest justifiable distance. While it is easier to detect and accurately estimate a larger change, larger changes are often associated with a higher-sensitivity MLE, requiring more noise (and therefore additional error) or truncation (and therefore information loss) to preserve privacy. We propose that practitioners using our private change-point detection algorithm choose input hypotheses accordingly. This practical setting is considered in our numerical studies, presented in Section 2.5.

2.3.1 Finite Sample Accuracy Guarantees for the MLE

Here we provide two accuracy bounds for the standard (non-private) MLE. These are the first finite-sample accuracy guarantees for this estimator. Such non-asymptotic properties have not been previously studied in traditional statistics, which typically focuses on consistency and unbiasedness of the estimator, with less attention to the convergence rate. We show that the additive error of the MLE is constant with respect to the sample size, which means that the convergence rate is $O_P(1)$. These results provide a baseline for quantifying the cost of privacy, since the techniques used in the theorem below mirror those used later in the accuracy proofs for our private algorithms.

A technical challenge that arises in proving accuracy of the estimator is that the x_i are not identically distributed when the true change-point $k^* \in (1, n]$, and so the partial log-likelihood ratios $\ell(k)$ are dependent across k. Hence we need to investigate a sequence of $\ell(k)$ that may be neither independent nor identically distributed. Fortunately, the differences $\ell(k) - \ell(k+1) = \log \frac{P_1(x_k)}{P_0(x_k)}$ are piecewise i.i.d.; that is, the differences are i.i.d. before the change point k^*, and i.i.d. after the change point k^*. This property is key in our proof. Moreover, we show that we can divide the possible outputs of the algorithm into regions of doubling size with exponentially decreasing probability of being selected by the algorithm, resulting in accuracy bounds that are independent of the number of data points n.

Note that our first accuracy guarantee depends on two measures $\Delta(\ell)$ and C of the distances between distributions P_0 and P_1. Accuracy is best for distributions for which $\Delta(\ell)$ is small relative to KL-divergence, which is consistent with the intuition that larger changes are easier to detect but output sensitivity degrades the robustness of the estimator, harming accuracy. This will be true for our first private algorithm OFFLINEPCPD, whose accuracy is additionally harmed by the extra noise required to protect privacy when output sensitivity is higher.

This dependence on $\Delta(\ell)$ is not inherent, however. Allowing $\Delta(\ell)$ to be infinite pre-

cludes our use of the same concentration inequalities in obtaining the accuracy guarantee, but the main idea in the proof can be salvaged by decomposing the change from P_0 to P_1 into a change from P_0 to the average distribution $(P_0 + P_1)/2$ and then the average distribution to P_1. Correspondingly, our second accuracy guarantee will use the alternative distance measure

$$C_M = \min\left\{D_{KL}\left(P_0||\frac{P_0+P_1}{2}\right), D_{KL}\left(P_1||\frac{P_0+P_1}{2}\right)\right\},$$

which will allow us to provide an MLE accuracy guarantee for arbitrary distributions.

Theorem 10. *For n data points drawn from P_0, P_1 such that $\Delta(\ell) < \infty$ with true change time $k^* \in (1, n]$, the MLE $\hat{k}$ is (α, β)-accurate for any $\beta > 0$ and*

$$\alpha = \frac{2\Delta(\ell)^2}{C^2}\log\frac{32}{3\beta}. \tag{2.7}$$

For n data points drawn from arbitrary P_0, P_1 with true change time $k^* \in (1, n)$, the MLE $\hat{k}$ is (α, β)-accurate for any $\beta > 0$ and

$$\alpha = \frac{35}{C_M^2}\log\frac{32}{3\beta}, \tag{2.8}$$

where C and C_M are defined in (2.3) and (2.4), respectively.

Proof. Given some true change-point k^* and error tolerance $\alpha > 0$, we can partition the set of bad possible outputs $\hat{k}$ into sub-intervals of exponentially increasing size as follows. For $i \geq 1$, let

$$\begin{aligned} R_i^- &= [k^* - 2^i\alpha, k^* - 2^{i-1}\alpha), \\ R_i^+ &= (k^* + 2^{i-1}\alpha, k^* + 2^i\alpha], \text{ and} \\ R_i &= R_i^- \cup R_i^+. \end{aligned}$$

Then we can bound the probability of the bad event as follows:

$$\beta = \Pr\left[|\hat{k} - k^*| > \alpha\right] \leq \sum_{i \geq 1} \Pr\left[\max_{k \in R_i}\{\ell(k) - \ell(k^*)\} > 0\right]. \tag{2.9}$$

This requires us to reason about the probability that the log-likelihood ratios for the data are not too far away from their expectation. Although the $\ell(k)$ are not independent across k, their pairwise differences $\ell(k+1) - \ell(k)$ are. When $\Delta(\ell) < \infty$ we can apply our corollary of Ottaviani's inequality (Corollary 2) to bound the probability that $\ell(k)$ exceeds $\ell(k^*)$ by appropriately defining several random variables corresponding to a data stream X drawn according to the change-point model.

Specifically, we can decompose the empirical log-likelihood difference between the true change-point k^* and any candidate k into the expected value of this difference and the sum of i.i.d. random variables with mean zero as follows:

$$U_j := \begin{cases} -\log \frac{P_0(x_j)}{P_1(x_j)} + D_{KL}(P_0||P_1), & j < k^* \\ -\log \frac{P_1(x_j)}{P_0(x_j)} + D_{KL}(P_1||P_0), & j \geq k^* \end{cases}$$

$$\ell(k) - \ell(k^*) = \begin{cases} \sum_{j=k}^{k^*-1} U_j - (k^* - k) D_{KL}(P_0||P_1), & k < k^* \\ \sum_{j=k^*}^{k-1} U_j - (k - k^*) D_{KL}(P_1||P_0), & k \geq k^* \end{cases}$$

We can rewrite $\ell(k) - \ell(k^*)$ as a zero-mean random variable subtracting a positive deterministic quantity $|k^* - k| D_{KL}(P_0||P_1)$. In other words, $\ell(k) - \ell(k^*)$ is positive only if the zero-mean random variable takes a sufficiently large positive value, which we will prove is a rare event. In this problem, we have to study the event $\{\max_k\{\ell(k) - \ell(k^*)\} > 0\}$, where $\max$ is taken over a suitable set. To this end, we introduce a sequence of random

variables defined as:

$$S_m = \begin{cases} \sum_{k^*+m \le j < k^*} U_j, & m < 0 \\ \sum_{k^* \le j < k^*+m} U_j & m > 0. \end{cases}$$

With these random variables, we bound the probability that the MLE lives in any particular bad subinterval $R_i, i \geq 1$ as follows:

$$\begin{aligned}
&\Pr\left[\max_{k \in R_i} \{\ell(k) - \ell(k^*)\} > 0\right] \\
&= \Pr\left[\max_{k \in R_i^-} \left\{\sum_{j=k}^{k^*-1} U_j - (k^* - k) D_{KL}(P_0||P_1)\right\} > 0\right] \\
&\quad + \Pr\left[\max_{k \in R_i^+} \left\{\sum_{j=k^*}^{k-1} U_j - (k - k^*) D_{KL}(P_1||P_0)\right\} > 0\right] \\
&= \Pr\left[\max_{k \in R_i^-} \left\{\sum_{j=k}^{k^*-1} U_j > (k^* - k) D_{KL}(P_0||P_1)\right\}\right] \\
&\quad + \Pr\left[\max_{k \in R_i^+} \left\{\sum_{j=k^*}^{k-1} U_j > (k - k^*) D_{KL}(P_1||P_0)\right\}\right] \\
&\leq \Pr\left[\max_{k \in [2^{i-1}\alpha]} |S_{-k}| > 2^{i-1}\alpha C\right] + \Pr\left[\max_{k \in [2^{i-1}\alpha]} |S_k| > 2^{i-1}\alpha C\right] \\
&\leq \frac{4 \cdot \exp(-2^{i-2}\alpha C^2/\Delta(\ell)^2)}{1 - 2 \cdot \exp(-2^{i-2}\alpha C^2/\Delta(\ell)^2)} \qquad (2.10) \\
&\leq 8 \exp(-2^{i-2}\alpha C^2/\Delta(\ell)^2) \qquad (2.11) \\
&= 8\left(\exp(\frac{-\alpha C^2}{2\Delta(\ell)^2})\right)^{2^{i-1}},
\end{aligned}$$

where the first inequality comes from the definitions of R_i and C, inequality (2.10) follows from an application of Corollary 2 with $\lambda_1 = \lambda_2 = 2^{i-2}\alpha C$ and $L = \Delta(\ell)$, and the denominator can be simplified as in (2.11) under the assumption that $\alpha \geq \frac{2\Delta(\ell)^2 \log 4}{C^2}$, which is satisfied by the final bound on α in (2.7).

We now consider the sum of these terms over all i, which will be needed for the final

bound on Equation (2.9). We note that this sum is bounded above by a geometric series with ratio $\exp(-\alpha C^2/(2\Delta(\ell)^2))$ since $2^{i-1} \geq i$, yielding the second and third inequalities. For the fourth inequality, the same assumed lower bound on α is used to simplify the denominator as in (2.11):

$$\begin{aligned}
\sum_{i\geq 1} \Pr\left[\max_{k\in R_i}\{\ell(k)-\ell(k^*)\} > 0\right] &\leq 8\sum_{i\geq 1}\left(\exp(\frac{-\alpha C^2}{2\Delta(\ell)^2})\right)^{2^{i-1}} \\
&\leq 8\sum_{i\geq 1}\left(\exp(\frac{-\alpha C^2}{2\Delta(\ell)^2})\right)^{i} \qquad (2.12)\\
&\leq \frac{8\exp(\frac{-\alpha C^2}{2\Delta(\ell)^2})}{1-\exp(\frac{-\alpha C^2}{2\Delta(\ell)^2})} \qquad (2.13)\\
&\leq \frac{32}{3}\exp\left(\frac{-\alpha C^2}{2\Delta(\ell)^2}\right). \qquad (2.14)
\end{aligned}$$

Equations (2.12)–(2.14) involve a standard technique in bounding geometric series. This technique will be invoked several times throughout the remainder of this paper. When it is used again in future proofs, we will omit the intermediate steps and only show the final results. For α as in (2.7) in the theorem statement, the expression above is bounded by β as required.

In the case that $\Delta(\ell)$ is infinite, we instead define i.i.d. random variables V_j with mean zero, according to an alternative log-likelihood as follows:

$$V_j := \begin{cases} -\log\frac{P_0(x_j)}{(\frac{P_0+P_1}{2})(x_j)} + D_{KL}\left(P_0||\frac{P_0+P_1}{2}\right), & j < k^* \\ -\log\frac{P_1(x_j)}{(\frac{P_0+P_1}{2})(x_j)} + D_{KL}\left(P_1||\frac{P_0+P_1}{2}\right), & j \geq k^* \end{cases}$$

This new set of random variables is necessary when $\Delta(\ell)$ is infinite, because the U_j no longer have bounded support, so we cannot apply Corollary 2. Instead we will apply a corollary of Bernstein's inequality (Corollary 4) to get similar bounds.

With these random variables, we can bound the empirical log-likelihood difference

between the true change-point k^* and any candidate k by,

$$\frac{1}{2}[\ell(k) - \ell(k^*)] \leq \begin{cases} \sum_{j=k}^{k^*-1} V_j - (k^* - k) D_{KL}\left(P_0 || \frac{P_0+P_1}{2}\right), & k < k^* \\ \sum_{j=k^*}^{k-1} V_j - (k - k^*) D_{KL}\left(P_1 || \frac{P_0+P_1}{2}\right), & k \geq k^*. \end{cases}$$

The inequality follows by concavity of the log function, which gives that $\frac{1}{2}\log\frac{P_1(x)}{P_0(x)} \leq \log\frac{0.5(P_0+P_1)(x)}{P_0(x)}$ for any x. Next we bound each term in (2.9) for any $i \geq 1$ as follows, noting that the 1/2 multiplier has no effect when we are only concerned with the maximum being positive:

$$\begin{aligned}
&\Pr\left[\max_{k\in R_i}\{\ell(k) - \ell(k^*)\} > 0\right] \\
&\quad\leq \Pr\left[\max_{k\in R_i^-}\left\{\sum_{j=k}^{k^*-1} V_j - (k^*-k) D_{KL}\left(P_0||\frac{P_0+P_1}{2}\right)\right\} > 0\right] \\
&\qquad + \Pr\left[\max_{k\in R_i^+}\left\{\sum_{j=k^*}^{k-1} V_j - (k-k^*) D_{KL}\left(P_1||\frac{P_0+P_1}{2}\right)\right\} > 0\right] \\
&\quad= \Pr\left[\max_{k\in R_i^-}\left\{\sum_{j=k}^{k^*-1} V_j > (k^*-k) D_{KL}\left(P_0||\frac{P_0+P_1}{2}\right)\right\}\right] \\
&\qquad + \Pr\left[\max_{k\in R_i^+}\left\{\sum_{j=k^*}^{k-1} V_j > (k-k^*) D_{KL}\left(P_1||\frac{P_0+P_1}{2}\right)\right\}\right] \\
&\quad\leq \Pr\left[\max_{k\in[2^{i-1}\alpha]} |\textstyle\sum_{j=k^*-k}^{k^*-1} V_j| > 2^{i-1}\alpha C_M\right] + \Pr\left[\max_{k\in[2^{i-1}\alpha]} |\textstyle\sum_{j=k^*}^{k^*+k-1} V_j| > 2^{i-1}\alpha C_M\right] \qquad (2.15) \\
&\quad\leq \frac{4\exp\left(-\frac{2^{i-3}\alpha C_M^2}{C_M+8}\right)}{1 - 2\exp\left(-\frac{2^{i-3}\alpha C_M^2}{C_M+8}\right)} \qquad (2.16) \\
&\quad\leq 8\exp\left(-\frac{2^{i-3}\alpha C_M^2}{C_M+8}\right), \qquad (2.17)
\end{aligned}$$

where (2.15) follows from the definitions of R_i and C_M, and (2.16) follows from an application of Corollary 4 with $\lambda_1 = \lambda_2 = 2^{i-2}\alpha C_M$ and $v = 4$. The denominator is simplified in (2.17) using our final bound on α in (2.8) and direct calculations to show that

$\alpha \geq \frac{35}{C_M^2} \log \frac{32}{3\beta} > 82/C_M^2$ implies $2\exp\left(-\frac{2^{i-3}\alpha C_M^2}{C_M+8}\right) < 1/2$.

To verify the application of Corollary 4 used in Equation (2.16), we need to show that for all j,

$$\mathbb{E}\left[\exp(|V_j|) - 1 - |V_j|\right] \leq 2. \tag{2.18}$$

To show this, let Y_j be the biased i.i.d. alternative log-likelihood ratio as follows:

$$Y_j = \begin{cases} \frac{\left(\frac{P_0+P_1}{2}\right)(x_j)}{P_0(x_j)}, & j < k^* \\ \frac{\left(\frac{P_0+P_1}{2}\right)(x_j)}{P_1(x_j)}, & j \geq k^* \end{cases}$$

Because $P_i(x)/\left(\frac{P_0+P_1}{2}\right)(x) \leq 2$ for $i = 0, 1$, we have

$$0 \leq D_{KL}(P_i||(P_0+P_1)/2) \leq \log 2, \tag{2.19}$$

and thus $e^{D_{KL}(P_i||(P_0+P_1)/2)} \in [1, 2]$. It suffices to note that $\mathbb{E}\left[\exp(|V_j|)\right] \leq 3$, because $\mathbb{E}\left[\exp(|V_j|) - 1 - |V_j|\right] \leq \mathbb{E}\left[\exp(|V_j|) - 1\right]$. We present the analysis when $j < k^*$, and the following expectation is taken under P_0. Note that the other side $j \geq k^*$ is similar with the expectation taken under P_1.

$$\begin{aligned}
\mathbb{E}\left[\exp(|V_j|)\right] &= \mathbb{E}\left[\exp(|\log Y_j - \mathbb{E}\left[\log Y_j\right]|)\right] \\
&\leq \mathbb{E}\left[\exp(\log Y_j - \mathbb{E}\left[\log Y_j\right])\right] + \mathbb{E}\left[\exp(\mathbb{E}\left[\log Y_j\right] - \log Y_j)\right] \\
&= \mathbb{E}\left[Y_j\right] e^{D_{KL}(P_0||(P_0+P_1)/2)} + \frac{\mathbb{E}\left[1/Y_j\right]}{e^{D_{KL}(P_0||(P_0+P_1)/2)}} \\
&\leq e^{D_{KL}(P_0||(P_0+P_1)/2)} + \frac{2}{e^{D_{KL}(P_0||(P_0+P_1)/2)}} \qquad (2.20) \\
&\leq 2\sqrt{2} \leq 3, \qquad (2.21)
\end{aligned}$$

where (2.20) follows from $\mathbb{E}\left[Y_j\right] = 1$, and $\mathbb{E}\left[1/Y_j\right] = \mathbb{E}\left[P_0(x)/(\frac{P_0+P_1}{2})(x)\right] \leq 2$, and (2.21) follows from the optimization that $x + 2/x \leq 2\sqrt{2}$ for $x \in [1, 3]$.

Finally, we consider the sum of the terms (2.17) over all i:

$$\sum_{i\geq 1} \Pr\left[\max_{k\in R_i}\{\ell(k)-\ell(k^*)\}>0\right] \leq \sum_{i\geq 1} 8\left(\exp\left(-\frac{\alpha C_M^2}{4(C_M+8)}\right)\right)^{2^{i-1}}$$

$$\leq \frac{8\exp\left(-\frac{\alpha C_M^2}{4(C_M+8)}\right)}{1-\exp\left(-\frac{\alpha C_M^2}{4(C_M+8)}\right)} \tag{2.22}$$

$$\leq \frac{32}{3}\exp\left(-\frac{\alpha C_M^2}{35}\right), \tag{2.23}$$

where inequality (2.22) uses the standard technique as in Equations (2.12)–(2.13). The denominator is simplified in (2.23) using the condition that $\exp\left(-\frac{\alpha C_M^2}{4(C_M+8)}\right) < 1/4$; we also use the fact that $C_M \leq \log 2$ according to (2.19). For α as in (2.8) in the theorem statement, the expression above is bounded by β, completing the proof.

□

2.3.2 Offline Algorithm under the Uniform Bound Assumption

Our first private offline algorithm OFFLINEPCPD applies the REPORTMAX algorithm [19] to the change-point problem by adding Laplace noise with parameter $\Delta(\ell)/\epsilon$ to each finite-sensitivity partial log-likelihood ratio $\ell(k)$ in order to estimate the private change-point. We note that our algorithm can be easily modified to additionally output an approximation of $\ell(\tilde{k})$ and incur 2ϵ privacy cost by composition.

Algorithm 4 Offline private change-point detector: OFFLINEPCPD$(X, P_0, P_1, \epsilon, n)$

Input: database X, distributions P_0, P_1, privacy parameters ϵ, database size n

Let $\Delta(\ell) = \max_x \ \log \frac{P_1(x)}{P_0(x)} - \min_{x'} \ \log \frac{P_1(x')}{P_0(x')}$

for $k = 1, \ldots, n$ **do**

Compute $\ell(k) = \sum_{i=k}^{n} \log \frac{P_1(x_i)}{P_0(x_i)}$

Sample $Z_k \sim \text{Lap}(\frac{\Delta(\ell)}{\epsilon})$

end for

Output $\tilde{k} = \underset{1 \le k \le n}{\text{argmax}} \{\ell(k) + Z_k\}$

Privacy of OFFLINEPCPD follows by instantiation of REPORTMAX [19] with queries $\ell(k)$ for $k \in [n]$, which have sensitivity $\Delta(\ell)$; this proof is included for completeness.

Theorem 11. *For arbitrary data X and $\epsilon > 0$,* OFFLINEPCPD(X, P_0, P_1, ϵ) *is ϵ-differentially private.*

Proof. Fix any two neighboring databases X, X' that differ on index j. For any $k \in [n]$, denote the respective partial log-likelihood ratios as $\ell(k)$ and $\ell'(k)$. By (2.1), we have

$$\ell'(k) = \ell(k) + \Delta \mathbb{I}\{j \ge k\} \qquad \text{with} \quad \Delta = \log \frac{P_1(x'_j)}{P_0(x'_j)} - \log \frac{P_1(x_j)}{P_0(x_j)}. \tag{2.24}$$

Next, for a given $1 \le i \le n$, fix Z_{-i}, a draw from $[\text{Lap}(\Delta(\ell)/\epsilon)]^{n-1}$ used for all the noisy log likelihood ratio values except the ith one. We will bound from above and below the ratio of the probabilities that the algorithm outputs $\tilde{k} = i$ on inputs X and X'. Define the minimum noisy value in order for i to be select with X:

$$Z_i^* = \min\{Z_i : \ell(i) + Z_i > \ell(k) + Z_k \quad \forall k \ne i\}$$

If $\Delta < 0$, then for all $k \ne i$ we have

$$\ell'(i) + \Delta(\ell) + Z_i^* \ge \ell(i) + Z_i^* > \ell(k) + Z_k \ge \ell'(k) + Z_k.$$

If $\Delta \geq 0$, then for all $k \neq i$ we have

$$\ell'(i) + Z_i^* \geq \ell(i) + Z_i^* > \ell(k) + Z_k \geq \ell'(k) - \Delta(\ell) + Z_k.$$

Hence, $Z_i' \geq Z_i^* + \Delta(\ell)$ ensures that the algorithm outputs i on input X', and the theorem follows from the following inequalities for any fixed Z_{-i}, with probabilities over the choice of $Z_i \sim \text{Lap}(\Delta(\ell)/\epsilon)$.

$$\Pr[\tilde{k} = i \mid X', Z_{-i}] \geq \Pr[Z_i' \geq Z_i^* + A \mid Z_{-i}] \geq e^{-\epsilon} \Pr[Z_i \geq Z_i^* \mid Z_{-i}] = e^{-\epsilon} \Pr[\tilde{k} = i \mid X, Z_{-i}].$$

□

Next we provide an accuracy guarantee for the output $\tilde{k}$ of our private algorithm OFFLINEPCPD when the data are drawn from P_0, P_1 with true change point $k^* \in (1, n)$. By providing this bound using a technique mirroring that of Theorem 10 to bound the error of the non-private MLE, Theorem 12 quantifies the marginal cost of requiring privacy in change-point detection. This additional cost comes from the fact that not only may the randomness of the n data points X result in an incorrect MLE, but the randomness of the Laplace noise added for privacy may also result in an incorrect noisy estimate of the MLE.

Theorem 12. *For hypotheses P_0, P_1 such that $\Delta(\ell) < \infty$ and n data points X drawn from P_0, P_1 with true change time $k^* \in (1, n]$, and for privacy parameter $\epsilon > 0$, the algorithm* OFFLINEPCPD$(X, P_0, P_1, \epsilon, n)$ *is (α, β)-accurate for any $\beta > 0$ and*

$$\alpha = \max\left\{ \frac{8\Delta(\ell)^2}{C^2} \log\frac{64}{3\beta}, \frac{8\Delta(l)}{C\epsilon} \log\frac{64\Delta(l)}{\beta C\epsilon} \right\}. \tag{2.25}$$

Proof. Our proof is structured around the observation that the algorithm only outputs a particular incorrect $\tilde{k} \neq k^*$ if there exists some k in which $\ell(k) + Z_k > \ell(k^*) + Z_{k^*}$ for

a set of random noise values $\{Z_k\}_{k\in[n]}$ selected by the algorithm. For the algorithm to output an incorrect value, there must either be a k that nearly beats the true change point on the noiseless data or there must be a k that receives much more noise than k^*. Intuitively, this captures the respective scenarios that unusual data causes non-private ERM to perform poorly and that unusual noise draws causes our private algorithm to perform poorly.

As in the proof of Theorem 10, given some true change-point k^* and error tolerance $\alpha > 0$, we partition the set of bad possible outputs k into sub-intervals of exponentially increasing size. For $i \geq 1$, let:

$$\begin{aligned} R_i^- &= [k^* - 2^i\alpha, k^* - 2^{i-1}\alpha), \\ R_i^+ &= (k^* + 2^{i-1}\alpha, k^* + 2^i\alpha], \text{ and} \\ R_i &= R_i^- \cup R_i^+. \end{aligned}$$

Then for any range-specific thresholds t_i for $i \geq 1$, our previous observations allow us to bound the probability of the bad event as follows:

$$\beta = \Pr[|\tilde{k} - k^*| > \alpha] \leq \sum_{i\geq 1} \Pr\left[\max_{k\in R_i}\{\ell(k) - \ell(k^*)\} > -t_i\right] + \sum_{i\geq 1} \Pr\left[\max_{k\in R_i}\{Z_k - Z_{k^*}\} \geq t_i\right] \tag{2.26}$$

We bound each term in the above expression separately for $t_i = 2^{i-2}\alpha C$, and we will set α to ensure that each term is at most $\beta/2$. As in Theorem 10, we can bound the first set of terms using Corollary 2 to bound the probability that $\ell(k)$ significantly exceeds $\ell(k^*)$ by appropriately defining several random variables corresponding to a data stream X drawn according to the change-point model:

$$U_j = \begin{cases} -\log \frac{P_0(x_j)}{P_1(x_j)} + D_{KL}(P_0||P_1), & j < k^* \\ -\log \frac{P_1(x_j)}{P_0(x_j)} + D_{KL}(P_1||P_0), & j \geq k^* \end{cases}$$

$$\ell(k) - \ell(k^*) = \begin{cases} \sum_{j=k}^{k^*-1} U_j - (k^* - k) D_{KL}(P_0||P_1), & k < k^* \\ \sum_{j=k^*}^{k-1} U_j - (k - k^*) D_{KL}(P_1||P_0), & k \geq k^* \end{cases}$$

We also define random variable S_m to denote the sum of m i.i.d. random variables as follows, noting that S_m is distributed like $\sum_{j=k^*+m}^{k^*-1} U_j$ for $m < 0$ and like $\sum_{j=k^*}^{k^*+m-1} U_j$ for $m > 0$.

$$S_m = \begin{cases} \sum_{k^*+m \leq j < k^*} U_j, & m < 0 \\ \sum_{k^* \leq j < k^*+m} U_j & m > 0 \end{cases}$$

With these random variables, we bound each term in the first set of terms in (2.26) for any $i \geq 1$ and threshold $t_i = 2^{i-2}\alpha C$ as follows:

$$\begin{aligned} &\Pr\left[\max_{k \in R_i} \{\ell(k) - \ell(k^*)\} > -2^{i-2}\alpha C\right] \\ &\leq \Pr\left[\max_{k \in R_i^-} \left\{\sum_{j=k}^{k^*-1} U_j - (k^* - k) D_{KL}(P_0||P_1)\right\} > -2^{i-2}\alpha C\right] \\ &\quad + \Pr\left[\max_{k \in R_i^+} \left\{\sum_{j=k^*}^{k-1} U_j - (k - k^*) D_{KL}(P_1||P_0)\right\} > -2^{i-2}\alpha C\right] \\ &\leq \Pr\left[\max_{k \in [2^{i-1}\alpha]} |S_{-k}| > 2^{i-2}\alpha C\right] + \Pr\left[\max_{k \in [2^{i-1}\alpha]} |S_k| > 2^{i-2}\alpha C\right] \\ &\leq \frac{4 \cdot \exp(-2^{i-4}\alpha C^2/\Delta(\ell)^2)}{1 - 2 \cdot \exp(-2^{i-4}\alpha C^2/\Delta(\ell)^2)} \qquad (2.27) \\ &\leq 8 \exp(-2^{i-4}\alpha C^2/\Delta(\ell)^2) \qquad (2.28) \\ &= 8 \left(\exp(\frac{-\alpha C^2}{8\Delta(\ell)^2})\right)^{2^{i-1}}, \end{aligned}$$

where (2.27) follows from an application of Corollary 2 with $\lambda_1 = \lambda_2 = 2^{i-3}\alpha C$ and $L = \Delta(\ell)$, and the denominator can be simplified as in (2.28) under the assumption that $\alpha \geq \frac{8\Delta(\ell)^2 \log 4}{C^2}$, which is satisfied by our final bounds.

We now consider the sum of these terms over all i, which will be needed for the final bound on Equation (2.26). The same technique that was used to bound geometric series in Equations (2.12)–(2.14) is applied here. For the fourth inequality, the same assumed lower bound on α is used to simplify the denominator as in (2.28):

$$\begin{aligned}
\sum_{i\geq 1} \Pr\left[\max_{k\in R_i}\{\ell(k) - \ell(k^*)\} > -2^{i-2}\alpha C\right] &\leq 8\sum_{i\geq 1}\left(\exp(\frac{-\alpha C^2}{8\Delta(\ell)^2})\right)^{2^{i-1}} \\
&\leq \frac{8\exp(\frac{-\alpha C^2}{8\Delta(\ell)^2})}{1-\exp(\frac{-\alpha C^2}{8\Delta(\ell)^2})} \\
&\leq \frac{32}{3}\exp\left(\frac{-\alpha C^2}{8\Delta(\ell)^2}\right). \qquad (2.29)
\end{aligned}$$

The first term in (2.25) in the theorem statement ensures that the expression above is bounded by $\beta/2$. It remains to show that the second term in (2.25) is enough to guarantee that the Laplace noise added for privacy will not harm accuracy except with probability $\beta/2$.

Next we bound the second set of terms of (2.26). We can easily bound one term in this set for any $i \geq 1$ since each Z_k and Z_{k^*} are independent draws from a Laplace distribution with parameter $\Delta(\ell)/\epsilon$, allowing us to apply a union bound over all indices in R_i:

$$\begin{aligned}
\Pr\left[\max_{k\in R_i}\{Z_k - Z_{k^*}\} \geq 2^{i-2}\alpha C\right] &\leq \Pr\left[2\max_{k\in R_i}|Z_k| \geq 2^{i-2}\alpha C\right] \\
&\leq 2^i\alpha\Pr[|\text{Lap}(\Delta(\ell)/\epsilon)| \geq 2^{i-3}\alpha C] \\
&\leq 2^i\alpha\cdot\exp\left(\frac{-2^{i-3}\alpha C\epsilon}{\Delta(\ell)}\right) \\
&= 2^i\alpha\left(\exp\left(\frac{-\alpha C\epsilon}{4\Delta(\ell)}\right)\right)^{2^{i-1}}.
\end{aligned}$$

Then by summing over all ranges and assuming in (2.30) that $\alpha \geq \frac{4\Delta(\ell)\log 2}{C\epsilon}$ to simplify the denominator, which will be satisfied by our final bound on α, we obtain a bound on the probability of large noise applied to any possible k far from k^*.

$$\begin{aligned}
\sum_{i\geq 1} \Pr\left[\max_{k\in R_i}\{Z_k - Z_{k^*}\} > 2^{i-2}\alpha C\right] &\leq \alpha \sum_{i\geq 1} 2^i \left(\exp(\frac{-\alpha C\epsilon}{4\Delta(\ell)})\right)^{2^{i-1}} \\
&\leq 2\alpha \sum_{i\geq 1} i \left(\exp(\frac{-\alpha C\epsilon}{4\Delta(\ell)})\right)^{i} \\
&= 2\alpha \frac{\exp(\frac{-\alpha C\epsilon}{4\Delta(\ell)})}{(1-\exp(\frac{-\alpha C\epsilon}{4\Delta(\ell)}))^2} \\
&\leq 8\alpha \exp\left(\frac{-\alpha C\epsilon}{4\Delta(\ell)}\right). \qquad (2.30)
\end{aligned}$$

To ensure that (2.30) is at most $\beta/2$, we require that $\log\alpha - \frac{\alpha C\epsilon}{4\Delta(l)} \leq \log\frac{\beta}{16}$. Adding $\log\frac{C\epsilon}{4\Delta(l)}$ to both sides, we have $\log\frac{\alpha C\epsilon}{4\Delta(l)} - \frac{\alpha C\epsilon}{4\Delta(l)} \leq \log\frac{\beta C\epsilon}{64\Delta(l)}$. Since $\frac{\alpha C\epsilon}{8\Delta(l)} \geq \log\frac{\alpha C\epsilon}{4\Delta(l)}$, requiring $-\frac{\alpha C\epsilon}{8\Delta(l)} \leq \log\frac{\beta C\epsilon}{64\Delta(l)}$, or equivalently $\alpha \geq \frac{8\Delta(l)}{C\epsilon}\log\frac{64\Delta(l)}{\beta C\epsilon}$ suffices to bound (2.30) above by $\beta/2$.

□

2.3.3 Offline Algorithm for Arbitrary Distributions

In this subsection, we give an offline private change-point detector OFFLINEPTCPD that offers guarantees even when $\Delta(\ell)$ is infinite. Relaxing the uniform bound assumption means that we may have a single data point x_j that dramatically increases $\ell(k)$ for $k \geq j$, so we cannot add noise proportional to $\Delta(\ell)$. Instead we truncate the log-likelihood ratio and add noise proportional to the post-truncation range. We compute the A-truncated log-likelihood ratio of k as

$$\ell_A(k) = \sum_{i=k}^{n} \left[\log\frac{P_1(x_i)}{P_0(x_i)}\right]_{-A/2}^{A/2},$$

where $[x]_a^b$ denotes the projection of x onto the interval $[a, b]$. This truncation scheme yields privacy immediately by instantiation of REPORTMAX [19] with queries $\ell_A(k)$ for $k \in [n]$, which have sensitivity $\Delta(\ell_A) = A$.

Algorithm 5 Offline private change-point detector: OFFLINEPTCPD($X, P_0, P_1, \epsilon, n, A$)

Input: database X, distributions P_0, P_1, privacy parameter ϵ, database size n, truncation parameter A

for $k = 1, \ldots, n$ **do**

Compute $\ell_A(k) = \sum_{i=k}^{n} \left[\log \frac{P_1(x_i)}{P_0(x_i)}\right]_{-A/2}^{A/2}$

Sample $Z_k \sim \text{Lap}(\frac{A}{\epsilon})$

end for

Output $\tilde{k} = \underset{1 \le k \le n}{\text{argmax}}\{\ell_A(k) + Z_k\}$ ▷ Report noisy argmax

Theorem 13. *For arbitrary data X and $\epsilon > 0$, OFFLINEPTCPD$(X, P_0, P_1, \epsilon, A)$ is ϵ-differentially private.*

Since we are no longer able to uniformly bound $P_1(x)/P_0(x)$, our accuracy results include a truncation parameter A in place of $\Delta(\ell)$ since A is the sensitivity of ℓ_A. Rather than C, the distributional difference measure parametrizing our results correspondingly depends on the truncation parameter A, which must be chosen to ensure C_A is positive:

$$C_A = \min\left\{-\mathbb{E}_{x\sim P_0}\left[\log\frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2}, \mathbb{E}_{x\sim P_1}\left[\log\frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2}\right\}.$$

We note that for Gaussian and Gamma distributions, any $A > 0$ ensures $C_A > 0$. In Section 2.5 we illustrate that for these distributions, it is best to choose small A to avoid excess noise and effectively rely on the sign of the log-likelihood ratio for accuracy. For general $P_0 \neq P_1$, $A > 2$ is a sufficient condition by the following argument. When $A > 2$, we have $[\log x]_{-A/2}^{A/2} \le x - 1$, and thus $\mathbb{E}_{P_0}\left[\log\frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2} < \mathbb{E}_{P_0}\left[\frac{P_1(x)}{P_0(x)} - 1\right] = 0$ and $\mathbb{E}_{P_1}\left[\log\frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2} = -\mathbb{E}_{P_0}\left[\log\frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2} > 0$.

With these definitions, we are ready to present the accuracy of OFFLINEPTCPD, in which the quantities A and $\mathbb{E}\left[\log \frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2}$ play roles analogous to $\ell(k)$ and D_{KL} in Theorem 12.

Theorem 14. *For arbitrary hypotheses P_0, P_1 and n data points X drawn from P_0, P_1 with true change time $k^* \in (1, n)$, for privacy parameter $\epsilon > 0$, and for truncation parameter A that satisfies $C_A > 0$,* OFFLINEPTCPD$(X, P_0, P_1, \epsilon, n, A)$ *is (α, β)-accurate for any $\beta > 0$ and*

$$\alpha = \max\left\{\frac{8A^2}{C_A^2}\log\frac{64}{3\beta},\ \frac{8A}{C_A\epsilon}\log\frac{64A}{\beta C_A\epsilon}\right\}, \tag{2.31}$$

where C_A is defined in (2.5).

Proof. Given some true change-point k^* and error tolerance $\alpha > 0$, we can partition the set of bad possible outputs k into sub-intervals of exponentially increasing size as follows. Following the notation of Theorem 12, for $i \geq 1$ we let

$$\begin{aligned} R_i^- &= [k^* - 2^i\alpha, k^* - 2^{i-1}\alpha), \\ R_i^+ &= (k^* + 2^{i-1}\alpha, k^* + 2^i\alpha], \text{ and} \\ R_i &= R_i^- \cup R_i^+, \end{aligned}$$

and for range-specific thresholds t_i for $i \geq 1$, we will bound the probability of a bad output as follows:

$$\Pr[|\tilde{k} - k^*| > \alpha] \leq \sum_{i\geq 1}\Pr[\max_{k\in R_i}\{\ell_A(k) - \ell_A(k^*)\} > t_i] + \sum_{i\geq 1}\Pr[\max_{k\in R_i}\{Z_k - Z_{k^*}\} \geq t_i]. \tag{2.32}$$

In order to do this, we decompose the truncated log-likelihood difference between the true change-point k^* and any candidate k into the sum of i.i.d. random variables with mean zero and the expected value of this difference as follows:

$$U_j = \begin{cases} \left[\log \frac{P_1(x_j)}{P_0(x_j)}\right]_{-A/2}^{A/2} - \mathbb{E}_{x\sim P_0}\left[\log \frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2}, & j < k^* \\ -\left[\log \frac{P_1(x_j)}{P_0(x_j)}\right]_{-A/2}^{A/2} + \mathbb{E}_{x\sim P_1}\left[\log \frac{P_1(x)}{P_0(x)}\right]_{-A/2}^{A/2}, & j \geq k^* \end{cases}$$

$$\ell_A(k) - \ell_A(k^*) = \begin{cases} \sum_{j=k}^{k^*-1} U_j + (k^* - k)\mathbb{E}_{x\sim P_0}\left[\log \frac{P_1(x)}{P_0(x_i)}\right]_{-A/2}^{A/2}, & k < k^* \\ \sum_{j=k^*}^{k-1} U_j - (k - k^*)\mathbb{E}_{x\sim P_1}\left[\log \frac{P_1(x)}{P_0(x_i)}\right]_{-A/2}^{A/2}, & k \geq k^* \end{cases}$$

The rest of the proof follows exactly as the proof of the accuracy of OFFLINEPCPD from Theorem 12 with ℓ replaced with ℓ_A, with $\Delta(\ell)$ replaced with A, and with C replaced with C_A. As before, we set $t_i = 2^{i-2}\alpha C_A$, and the constants are inherited exactly as is because the truncated log-likelihood is applicable to the concentration inequalities in the same way that the non-truncated but uniformly bounded log-likelihood was in Theorem 12.

□

2.4 Online Private Change-point Detection

In this section, we give new differentially private algorithms for change-point detection in the online setting. In this setting, the algorithm initially receives n data points $x_1, \ldots, x_n$ and then continues to receive data points one at a time. As before, the goal is to privately identify an approximation of the time k^* when the data change from distribution P_0 to P_1, and now we additionally want to identify this change shortly after it occurs. We first give an algorithm ONLINEPCPD for detecting a single change-point, and then we show how it can be extended to ONLINEPMCPD to detect multiple change-points. Our algorithms use OFFLINEPCPD as a subroutine, but can be modified in a straightforward way to use log-likelihood truncation and OFFLINEPTCPD if distributions do not satisfy the assumption of uniform boundedness.

2.4.1 Single Change-point

Even in the single change-point setting, our offline algorithm is not directly applicable because we do not know a priori how many points must arrive before a true change-point occurs. To resolve this, ONLINEPCPD works like ABOVETHRESH (presented in Section 2.2.2), determining after each new data entry arrives whether it is likely that a change occurred in the most recent n entries. When ONLINEPCPD at time j detects a sufficiently large (noisy) partial log-likelihood ratio $\ell(k, j) = \sum_{i=k}^{j} \log \frac{P_1(x_i)}{P_0(x_i)}$ for some k within n data points of j, it calls OFFLINEPCPD to privately determine the most likely change point $\tilde{k}$ in the window $\{x_{j-n+1}, \ldots, x_j\}$.

Privacy of ONLINEPCPD is immediate from composition of ABOVETHRESH and OFFLINEPCPD, each with privacy loss $\epsilon/2$. As before, accuracy requires X to be drawn from P_0, P_1 with some true change point k^*. This algorithm also requires a suitable choice of log-likelihood threshold T to guarantee that OFFLINEPCPD is called for a window of data that actually contains k^*. Specifically, T should be large enough that the algorithm is unlikely to call OFFLINEPCPD when $j < k^*$ but small enough so that it is likely to call OFFLINEPCPD by time $j = k^* + n/2$. When both of these conditions hold, we inherit the accuracy of OFFLINEPCPD.

With our final bounds, we note that $n \gg \frac{\Delta(\ell)}{C} \log(k^*/\beta)$ suffices for existence of a suitable threshold, and an analyst must have a reasonable approximation of k^* in order to choose such a threshold. Otherwise, the accuracy bound itself has no dependence on the change-point k^*.

Algorithm 6 Online private change-point detector: ONLINEPCPD$(X, P_0, P_1, \epsilon, n, T)$

Input: database X, distributions P_0, P_1, privacy parameter ϵ, starting size n, threshold T

Let $\Delta(\ell) = \max_x \log \frac{P_1(x)}{P_0(x)} - \min_{x'} \log \frac{P_1(x')}{P_0(x')}$

Let $\hat{T} = T + \text{Lap}(4\Delta(\ell)/\epsilon)$

for each new data point $x_j, j \geq n$ **do**

Compute $\ell_j = \max_{j-n+1 \leq k \leq j} \{\ell(k, j)\} = \max_{j-n+1 \leq k \leq j} \{\sum_{i=k}^{j} \log \frac{P_1(x_i)}{P_0(x_i)}\}$

Sample $Z_j \sim \text{Lap}(\frac{8\Delta(\ell)}{\epsilon})$

if $\ell_j + Z_j > \hat{T}$ **then**

Output $(j - n) +$ OFFLINEPCPD$(\{x_{j-n+1}, \ldots, x_j\}, P_0, P_1, \epsilon/2, n)$

Halt

end if

end for

Theorem 15. *For arbitrary data X and $\epsilon > 0$,* ONLINEPCPD$(X, P_0, P_1, \epsilon, n, T)$ *is ϵ-differentially private.*

Theorem 16. *For hypotheses P_0, P_1 such that $\Delta(\ell) < \infty$, a stream of data points X with starting size n drawn from P_0, P_1 with true change time $k^* \geq n/2$, privacy parameter $\epsilon > 0$, and threshold $T \in [T_L, T_U]$ with*

$$\begin{aligned} T_L &:= 2\Delta(\ell)\sqrt{2\log\frac{64k^*}{\beta}} - C + \frac{16\Delta(\ell)}{\epsilon}\log\frac{8k^*}{\beta}, \\ T_U &:= \frac{nC}{2} - \frac{\Delta(\ell)}{2}\sqrt{n\log(8/\beta)} - \frac{16\Delta(\ell)}{\epsilon}\log\frac{8k^*}{\beta}, \end{aligned}$$

we have that ONLINEPCPD$(X, P_0, P_1, \epsilon, n, T)$ *is (α, β)-accurate for any $\beta > 0$ and*

$$\alpha = \max\left\{\frac{8\Delta(\ell)^2}{C^2}\log\frac{128}{3\beta}, \frac{8\Delta(l)}{C\epsilon}\log\frac{128\Delta(l)}{\beta C\epsilon}\right\}.$$

In the above expressions, $C = \min\{D_{KL}(P_0||P_1), D_{KL}(P_1||P_0)\}$.

Proof. We first give a range $[T_L, T_U]$ of thresholds that ensure that except with probability $\beta/4$, the randomly sampled data stream satisfies the following two conditions for some α'. These conditions are inherited from the requirements for ABOVETHRESH accuracy, respectively capturing the requirements that the threshold is not reached too early and that it is reached at least by the time the window is centered around k^*:

1. For $T \geq T_L$, $\max_{k\in[j-n+1,j]} \ell(k,j) < T - \alpha'$ for every $j < k^*$.

2. For $T \leq T_U$, $\max_{k\in[k^*-n/2,k^*+n/2-1]} \ell(k, k+n/2) > T + \alpha'$.

When these conditions are satisfied, the ABOVETHRESH guarantee ensures that except with probability $\beta/4$, the randomness of the online algorithm ensures that it calls the offline algorithm on a window of data containing the true change-point. Then we will argue that our overall accuracy follows from the offline guarantee, where we will allow failure probability $\beta/2$.

We will get the first condition by taking a union bound over all windows tested before the change-point, of the probability that the maximum log-likelihood $\max_k \ell(k)$ for n elements $X = \{x_1, \ldots, x_n\}$ sampled from P_0 exceeds a given threshold. To bound this probability, we first define the following random variables.

$$U_j = -\log \frac{P_0(x_j)}{P_1(x_j)} + D_{KL}(P_0||P_1) \qquad\qquad S_m = \sum_{1\leq j\leq m} U_j$$

We note that each $\ell(k)$ is the sum of i.i.d. random variables, and that the maximum log-likelihood over m consecutive elements is equal in distribution to $\max_{k\in[m]} S_k - kD_{KL}(P_0||P_1)$. This yields the first inequality below. Inequality (2.33) comes from applying Corollary 2

with $\lambda_1 = \lambda_2 = 2^{i-2}C + t/2$ and interval length $L = \Delta(\ell)$.

$$\begin{aligned}
\Pr\left[\max_{1\le k\le n}\{\ell(k)\} > t\right] \le & \sum_{i\ge 1}\Pr\left[\max_{k\in[2^{i-1},2^i)}\{S_k - kD_{KL}(P_0||P_1)\} > t\right] \\
\le & \sum_{i\ge 1}\Pr\left[\max_{k\in[2^{i-1}]} S_k > 2^{i-1}C + t\right] \\
\le & \sum_{i\ge 1}\frac{2\exp(-(2^{i-2}C + t/2)^2/(2^{i-2}\Delta(\ell)^2))}{1 - 2\exp(-(2^{i-2}C + t/2)^2/(2^{i-2}\Delta(\ell)^2))} \qquad (2.33) \\
\le & 4\sum_{i\ge 1}\exp(-(2^{i-2}C + t/2)^2/(2^{i-2}\Delta(\ell)^2)) \qquad (2.34) \\
\le & 8\exp(-(2^{-1}C + t/2)^2/(2^{-1}\Delta(\ell)^2)) \qquad (2.35) \\
\le & \frac{\beta}{8k^*} \qquad (2.36)
\end{aligned}$$

Inequalities (2.34), (2.35), and (2.36) follow by plugging in $t = 2\Delta(\ell)\sqrt{2\log\frac{64k^*}{\beta}} - C$. This ensures that $1 - 2\exp(-(2^{i-2}C + t/2)^2/(2^{i-2}\Delta(\ell)^2)) \ge 1/2$, giving Inequality (2.34), and that the series is increasing exponentially in i, so we can collapse the sum with another factor of 2 by considering only $i = 1$ as in Inequality (2.35). Plugging in this same value of t to Inequality (2.35) also immediately gives Inequality (2.36). Taking a union bound over all the windows prior to the change-point, this shows that Condition 1 holds for $T_L = 2\Delta(\ell)\sqrt{2\log\frac{64k^*}{\beta}} - C + \alpha'$ except with probability $\beta/8$.

To show that the second condition holds except with additional probability $\beta/8$, we consider the window of data with the first half of data drawn from P_0 and the second half drawn from P_1 and bound the probability that $\ell(k^*)$ in this window is less than a given threshold as follows. We note that $\ell(k^*, k^* + n/2 - 1)$ is the sum of $n/2$ i.i.d. random variables $\log\frac{P_1(x_i)}{P_0(x_i)}$, although these variables are not mean-zero. Instead, we define mean-zero random variables $V_j = -\log\frac{P_1(x_j)}{P_0(x_j)} + D_{KL}(P_1||P_0)$, and write $\ell(k^*, k^* + n/2 - 1)$ in terms of these new variables, analogously to above. We can then bound the sum of the V_j

using Hoeffding's inequality to get Equation (2.37):

$$\begin{aligned}\Pr\left[\max_{k^*-n/2\le k<k^*+n/2}\{\ell(k,k^*+n/2-1)\}<t\right] &\le \Pr[\ell(k^*,k^*+n/2-1)<t] \\ &\le \Pr\left[\sum_{j=k^*,\ldots,k^*+n/2-1} V_j > \frac{nC}{2}-t\right] \\ &\le \exp(-4(nC/2-t)^2/(n\Delta(\ell)^2))\end{aligned} \tag{2.37}$$

Plugging in $t = \frac{nC}{2} - \frac{\Delta(\ell)}{2}\sqrt{n\log(8/\beta)}$ in this final expression ensures that (2.37) $\le \beta/8$. This ensures that Condition 2 is satisfied except with probability $\beta/8$ for $T_U = nC/2 - \Delta(\ell)\sqrt{n\log(8/\beta)} - \alpha'$.

Then we can instantiate the ABOVETHRESH accuracy guarantee with privacy parameter $\epsilon/2$ and accuracy parameter $\beta/4$ to ensure that for $\alpha' = \frac{16\Delta(\ell)\log(8k^*/\beta)}{\epsilon}$ when Conditions 1 and 2 are satisfied, ABOVETHRESH will identify a window containing the true change-point except with probability $\beta/4$. Combining this with the $\beta/4$ probability that Conditions 1 and 2 fail to hold when $T \in [T_L, T_U]$, we get that ONLINEPCPD calls OFFLINEPCPD in a window containing the change-point except with probability $\beta/2$ over the randomness of the data and of the online portion of the algorithm.

We next instantiate OFFLINEPCPD with appropriate parameters to ensure that conditioned on being called in the correct window, it will output a $\tilde{k}$ that is within α of the true change-point k^* with probability at most $\beta/2$. We can then complete the proof by taking a union bound over all the failure probabilities.

Our offline accuracy guarantee requires data points are sampled i.i.d. from P_0 before the change point and from P_1 thereafter. However, it remains to be shown that conditioning on the event that we call the offline algorithm in a correct window does not harm the accuracy guarantee too much. For a window size n, change-point k^*, stream X of at least $k^* + n/2$ data points, set of random coins required by ONLINEPCPD and its call to OFFLINEPCPD,

and a stopping index $\nu > n/2$, let $N(\nu)$ denote the event that ONLINEPCPD calls OFFLINEPCPD on a window centered at ν, i.e., $\{x_{\nu-n/2}, \ldots, x_{\nu+n/2-1}\}$, and let $F(\nu)$ denote the event that OFFLINEPCPD on the window centered at ν fails to output an approximation within α of k^*. Our previous arguments bound the probability of all $N(\nu)$ for ν outside of a good range $G = (k^* - n/2, k^*]$, and our offline guarantee bounds the probability of $F(\nu)$ for any $\nu \in G$ as long as the data are truly distributed according to the change-point model.

Failure of the online algorithm can be due to either failure to halt on a correct window or failure of the offline algorithm on a window containing the true change. Thus we can then bound the probability of failure of the online algorithm as:

$$\Pr[|\tilde{k} - k^*| > \alpha] \leq \sum_{\nu \notin G} \Pr[N(\nu)] + \Pr[\bigcup_{\nu \in G} F(\nu)]$$

The first summation is at most $\beta/2$ by our previous arguments on the accuracy of the online portion of the algorithm. It remains to calculate the second term. We can still partition the set of bad possible output into sub-intervals of exponentially increasing size as follows. For $i \geq 1$, let

$$\begin{aligned} R_i^- &= [k^* - 2^i\alpha, k^* - 2^{i-1}\alpha), \\ R_i^+ &= (k^* + 2^{i-1}\alpha, k^* + 2^i\alpha], \text{ and} \\ R_i &= R_i^- \cup R_i^+. \end{aligned}$$

Then we can bound the probability that the offline algorithm fails on any correct window

as:

$$\begin{aligned}
\Pr\left[\bigcup_{\nu\in G} F(\nu)\right] &\leq \Pr\left[\max_{\nu\in G}\left\{\max_{\substack{\nu-n/2\leq k\leq \nu+n/2-1\\ \text{s.t. } |k-k^*|>\alpha}} \{\ell(k,\nu+n/2-1)\right.\right. \\
&\qquad \left.\left. +Z_k - \ell(k^*,\nu+n/2-1) - Z_{k^*}\}\right\} > 0\right] \\
&= \Pr\left[\max_{\nu\in G}\left\{\max_{\substack{\nu-n/2\leq k\leq \nu+n/2-1\\ \text{s.t. } |k-k^*|>\alpha}} \sum_{j=k}^{\nu+n/2-1} \log\frac{P_1(x_j)}{P_0(x_j)}\right.\right. \\
&\qquad \left.\left. +Z_k - \sum_{j=k^*}^{\nu+n/2-1} \log\frac{P_1(x_j)}{P_0(x_j)} - Z_{k^*}\right\} > 0\right] \\
&= \Pr\left[\max_{\substack{k^*-n+1\leq k\leq k^*+n/2-1\\ \text{s.t. } |k-k^*|>\alpha}} \left\{\sum_{j=k}^{k^*} \log\frac{P_1(x_j)}{P_0(x_j)} + Z_k - Z_{k^*}\right\} > 0\right] \\
&\leq \sum_{i\geq 1} \Pr[\max_{k\in R_i}\{\sum_{j=k}^{k^*} \log\frac{P_1(x_j)}{P_0(x_j)}\} > -t_i] + \sum_{i\geq 1} \Pr[\max_{k\in R_i}\{Z_k - Z_{k^*}\} > t_i]
\end{aligned}$$

Notice that the final line above is identical to Equation (2.26) in the proof of Theorem 12 for the accuracy of OFFLINEPCPD: the first term is the empirical log-likelihood difference between the true change-point k^* and any candidate k, and the second term is difference between two independent draws of Laplace noise. Thus the remainder of the analysis follows that of Theorem 12 instantiated with parameters $\beta/2$ and $\epsilon/2$. This instantiation of Theorem 12 gives that $\Pr[\bigcup_{\nu\in G} F(\nu)]$ is also bounded by $\beta/2$ when $\alpha = \max\left\{\frac{8\Delta(\ell)^2}{C^2}\log\frac{128}{3\beta}, \frac{8\Delta(l)}{C\epsilon}\log\frac{128\Delta(l)}{\beta C\epsilon}\right\}$.

Combining this with our previous bound on the $N(\nu)$ terms, we get that $\Pr[|\tilde{k} - k^*| > \alpha] \leq \beta$ for the desired α value in the theorem statement.

□

2.4.2 Multiple Changes

We now show how to extend our ONLINEPCPD algorithm to detect multiple change-points. In this setting, the data change from distribution P_0 to P_1, from P_1 to P_2, ..., and from P_{m-1} to P_m at times k_1^*, k_2^*, ..., k_m^*, respectively. As data arrive, ONLINEPMCPD makes online determinations about when the current window is sufficiently likely to contain a change-point and calls OFFLINEPCPD when so. After each private report of a change-point $\tilde{k}_i$ the algorithm simply restarts the remaining stream of data points after the next n data points arrive and resumes scanning for subsequent change-points.

The idea of this algorithm is similar to the extension from ABOVETHRESH to SPARSE, but by assuming that the m change-points are separated pairwise by at least the starting database size n and by setting the thresholds to ensure that with high probability a change-point k_i^* is detected by time $k_i^* + n/2$, we can update our sliding window between change-point detections to ensure that each entry only participates in one call to ONLINEPCPD and we never miss a change-point. This means that privacy of ONLINEPMCPD is immediate from privacy of ONLINEPCPD and SPARSE, and the accuracy cost is only $\log m$ rather than $\sqrt{m}$.

Algorithm 7 Online private multiple change-point detector:
ONLINEPMCPD$(X, P_0, \ldots, P_m, \epsilon, n, T_1, \ldots, T_m)$

Input: database X, distributions $P_0, \ldots, P_m$, privacy parameter ϵ, starting size n, thresholds $T_1, \ldots, T_m$

Let $\Delta_1 = \max_x \log \frac{P_1(x)}{P_0(x)} - \min_{x'} \log \frac{P_1(x')}{P_0(x')}$

Let $\hat{T}_1 = T_1 + \text{Lap}(4\Delta_1/\epsilon)$

Let $i = 1$

for each new data point $x_j, j \geq n$ **do**

Compute $\ell_j = \max_{j-n+1 \leq k \leq j} \{\ell_i(k, j)\} = \max_{j-n+1 \leq k \leq j} \{\sum_{\iota=k}^{j} \log \frac{P_i(x_\iota)}{P_{i-1}(x_\iota)}\}$

Sample $Z_j \sim \text{Lap}(\frac{8\Delta_i}{\epsilon})$

if $\ell_j + Z_j > \hat{T}_i$ **then**

Output $\tilde{k}_i = (j - n) +$ OFFLINEPCPD$(\{x_{j-n+1}, \ldots, x_j\}, P_{i-1}, P_i, \epsilon/2, n)$

if $i = m$ **then**

Halt

else

Let $i = i + 1$

Let $\Delta_i = \max_x \log \frac{P_i(x)}{P_{i-1}(x)} - \min_{x'} \log \frac{P_i(x')}{P_{i-1}(x')}$

Let $\hat{T}_i = T_i + \text{Lap}(4\Delta_i/\epsilon)$

Wait for n new data points, i.e., let j advance by n

end if

end if

end for

Theorem 17. *For arbitrary data X and $\epsilon > 0$,* ONLINEPMCPD*$(X, P_0, \ldots, P_m, \epsilon, n, T_1, \ldots, T_m)$ is ϵ-differentially private.*

We do not incur privacy composition across the multiple runs of ONLINEPCPD because each subroutine runs on a disjoint database. After OFFLINEPCPD is called in the above algorithm, the algorithm waits for n new data points to arrive before beginning the

for loop again (which corresponds to starting the next instantiation of ONLINEPCPD). This means that none of the data points used in the algorithmic steps corresponding to the previous instantiation of ONLINEPCPD will be used in the next instantiation. Thus each instantiation of the ONLINEPCPD subroutine will operate on a disjoint subset of the database, so they together satisfy ϵ-differential privacy, and composition is not needed [50].

One might be concerned that the starting point of (and hence the database that is input to) the next instantiation of ONLINEPCPD inside ONLINEPMCPD depends on the halting time of the previous instantiation. In particular, it will be exactly n data points after the halting time of the previous instantiation. However, the halting time of ONLINEPCPD is computed in a differentially private manner, so any function of it—such as the halting time plus n—will be automatically differentially private by the post-processing guarantees of differential privacy.

It remains to prove accuracy for ONLINEPMCPD. As before, accuracy requires X to be drawn from P_0, P_1, …, P_m with some true change-points k_1^*, k_2^*, …, k_m^*. To detect each change-point k_i^*, the choice of log-likelihood threshold T_i may need to be modified according to the hypothesized distributions and possibly to the expected time until the next change-point, which depends on the accuracy of the previous output.

Theorem 18. *For hypotheses $P_0, P_1, \ldots, P_m$ such that $\Delta_i = \max_x \log \frac{P_i(x)}{P_{i-1}(x)} - \min_{x'} \log \frac{P_i(x')}{P_{i-1}(x')} < \infty$ for $i = 1, \ldots, m$, a stream of data points X with starting size n drawn from $P_0, P_1, \ldots, P_m$ with true change times $k_0^*, k_1^*, \ldots, k_m^*$ with $k_0^* = 0, k_1^* \geq n/2, k_i^* - k_{i-1}^* \geq 3n/2$ for $i = 2, \ldots, m$, privacy parameter $\epsilon > 0$, and thresholds $T_i \in [T_{L,i}, T_{U,i}]$ with*

$$\begin{aligned} T_{L,i} &:= 2\Delta_i \sqrt{2 \log \frac{64m(k_i^* - k_{i-1}^*)}{\beta}} - C_i + \frac{16\Delta_i}{\epsilon} \log \frac{8m(k_i^* - k_{i-1}^*)}{\beta}, \\ T_{U,i} &:= \frac{nC_i}{2} - \frac{\Delta_i}{2} \sqrt{n \log(8m/\beta)} - \frac{16\Delta_i}{\epsilon} \log \frac{8m(k_i^* - k_{i-1}^*)}{\beta} \end{aligned}$$

for $i = 1, \ldots, m$, we have that ONLINEPMCPD*$(X, P_0, \ldots, P_m, \epsilon, n, T_1, \ldots, T_m)$ is (α, β)-*

accurate for any $\beta > 0$ *and*

$$\alpha = \max\left\{\frac{8\Delta^2}{C^2}\log\frac{128m}{3\beta}, \frac{8\Delta(l)}{C\epsilon}\log\frac{128m\Delta(l)}{\beta C\epsilon}\right\}.$$

In the above expressions, $\Delta = \max\{\Delta_1, \ldots, \Delta_m\}$, $C_i = \min\{D_{KL}(P_{i-1}||P_i), D_{KL}(P_i||P_{i-1})\}$ *and* $C = \min\{C_1, \ldots, C_m\}$.

Proof. For α as in the theorem statement, we will decompose the probability that the algorithm fails to output α-approximations for every k_i^* into the sum of m conditional probabilities, each of which can be bounded by β/m by an instantiation of our accuracy theorem for OnlinePCPD. In the proof below, we let S_i for $i \in [m]$ denote the event that OnlinePMCPD calls OnlinePCPD for the ith time with k_i^* in the latter half of the window and OnlinePCPD outputs an α-approximation of k_i^*. Then we have that

$$\begin{aligned}
&\Pr[\textsc{OnlinePMCPD}(X, P_0, \ldots, P_m, \epsilon, n, T_1, \ldots, T_m) \text{ fails}] \\
&\leq \Pr[\bar{S}_1] + \sum_{i=2}^{m} \Pr[\bar{S}_i \cap S_1 \cap \cdots \cap S_{i-1}] \\
&\leq \Pr[\bar{S}_1] + \sum_{i=2}^{m} \Pr[\bar{S}_i \cap S_{i-1}] \\
&\leq \sum_{i\in[m]} \Pr[\textsc{OnlinePCPD}(X_i', P_{i-1}, P_i, \epsilon, n, T_i) \text{ fails}], \qquad (2.38)
\end{aligned}$$

for X_i' drawn according to the single change-point model with initial distribution P_{i-1} and post-change distribution P_i with change-point $k_i^* - k_{i-1}^*$. The third inequality is because the event $\bar{S}_i$ conditioned on $S_1 \cap \cdots \cap S_{i-1}$ is equivalent to the failure of OnlinePCPD on a data stream consistent with the single change-point model, and in particular, failure is most likely when there are as many data points drawn from P_{i-1} as possible. Then bounding each term follows from instantiation of the theorem for OnlinePCPD because we can treat the ending point of the previous detection window as the starting point of a new detection procedure.

To be more mathematically rigorous, for $i = 2, \ldots, m$, we have

$$\Pr[\bar{S}_i \cap S_{i-1}] \leq \Pr[\bar{S}_i | S_{i-1}] = \mathbb{E}[\Pr(\bar{S}_i | S_{i-1}, j_{i-1}) | S_{i-1}] = \mathbb{E}[\Pr(\bar{S}_i | j_{i-1}) | S_{i-1}], \quad (2.39)$$

where the last equality follows from the fact that S_{i-1} and $\bar{S}_i$ are independent conditional on j_{i-1}. This conditional independence is an immediate consequence of the fact that S_{i-1} depends only on the data $x_1, \ldots, x_{j_{i-1}}$ and $\bar{S}_i$ depends only on the data $x_{j_{i-1}+1}, x_{j_{i-1}+2}, \ldots$, which are mutually independent conditional on j_{i-1}, as j_{i-1} is a stopping time.

Our final goal is to bound $\Pr(\bar{S}_i | j_{i-1})$, which can be done by invoking Theorem 16. The only difference here is that the index of the first sample is $j_{i-1} + 1$, instead of 1. Thus we have to modify the upper and lower thresholds in Theorem 16. Define

$$T_{L,i}(j) := 2\Delta_i \sqrt{2 \log \frac{64m(k_i^* - j)}{\beta}} - C_i + \frac{16\Delta_i}{\epsilon} \log \frac{8m(k_i^* - j)}{\beta},$$

and

$$T_{U,i}(j) =: \frac{nC_i}{2} - \frac{\Delta_i}{2} \sqrt{n \log(8m/\beta)} - \frac{16\Delta_i}{\epsilon} \log \frac{8m(k_i^* - j)}{\beta}.$$

For $T_i \in [T_{L,i}, T_{U,i}]$ where

$$T_{L,i} = 2\Delta_i \sqrt{2 \log \frac{64m(k_i^* - k_{i-1}^*)}{\beta}} - C_i + \frac{16\Delta_i}{\epsilon} \log \frac{8m(k_i^* - k_{i-1}^*)}{\beta}$$

and

$$T_{U,i} = \frac{nC_i}{2} - \frac{\Delta_i}{2} \sqrt{n \log(8m/\beta)} - \frac{16\Delta_i}{\epsilon} \log \frac{8m(k_i^* - k_{i-1}^*)}{\beta},$$

we have $T_{L,i} \leq T_{L,i}(j) \leq T_i \leq T_{U,i}(j) \leq T_{U,i}(j)$ for any $j \in [k_{i-1}^*, k_{i-1}^* + n/2]$. Then by instantiation of Theorem 16, we have that $\Pr(\bar{S}_i | j_{i-1} = j) \leq \beta/m$ provided that $j \in [k_{i-1}^*, k_{i-1}^* + n/2]$. Note that the event S_{i-1} implies $j_{i-1} \in [k_{i-1}^*, k_{i-1}^* + n/2]$. Thus, $\Pr[\bar{S}_i \cap S_{i-1}]$ is bounded above by β/m, and (2.38) is bounded above by β.

□

2.5 Numerical Studies

In this section, we present results from Monte Carlo experiments designed to validate the theoretical results of previous sections. The theoretical privacy guarantees hold in the worst-case over all databases and over all outputs of the algorithm, so it is only necessary to empirically validate the accuracy of our algorithms. Our simulations consider both offline (Section 2.5.1) and online settings (Section 2.5.2) for the canonical problems of detecting a change in the mean of Bernoulli or Gaussian distributions. In the offline setting, we additionally show that our algorithms can accurately detect changes in the variance of Gaussian distribution and detect changes in the shape parameter of a Gamma distribution.

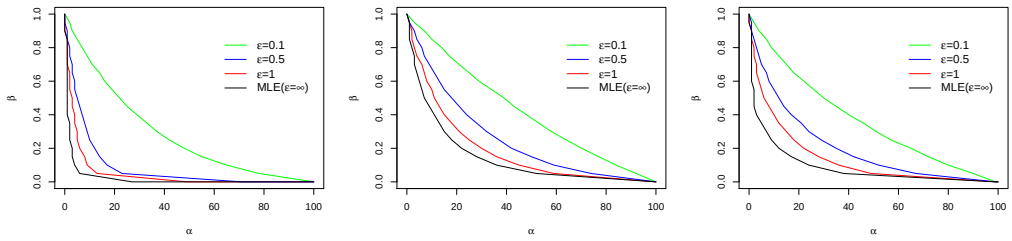
For completeness, we state the PMF of a Bernoulli distribution, and the PDF of Gaussian and Gamma distributions below.

- Bernoulli distribution: $\Pr(x = 1) = p$ and $\Pr(x = 0) = 1 - p$.
- Gaussian distribution: $f(x; \mu, \sigma) = (2\pi\sigma^2)^{1/2} \exp(-(x-\mu)^2/(2\sigma^2))$, where μ is the mean and σ is the standard deviation.
- Gamma distribution: $f(x; k, \theta) = (\Gamma(k)\theta^k)^{-1} x^{k-1} \exp(-x/\theta)$, where θ is the scale parameter and k is the shape parameter.

2.5.1 Evaluating the Offline Algorithms

Each simulation is characterized by a probability distribution family (Bernoulli, Gaussian, or Gamma), a distribution parameter that changes (mean, standard deviation, or shape), and a change magnitude (large, small, or underspecified). The large and small change regimes correspond respectively to large and small changes in the distribution parameter of interest. The underspecified regime corresponds to the setting where the true change is large, but the input parameters correspond to a small change. This setting goes beyond our theoretical results to suggest that our algorithm still performs well, even when the distributional parameters are misspecified. All parameters are stated in the caption.

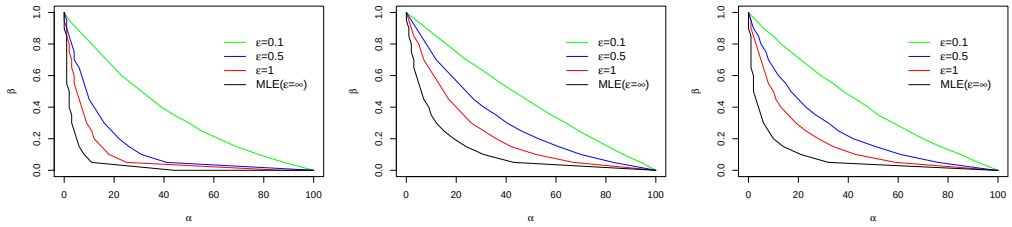
For Bernoulli distributions, the log-likelihood ratio is uniformly bounded and so we use OFFLINEPCPD; for Gaussian and Gamma distributions, we set $A = 0.1$ (for reasons discussed later in this section) and use OFFLINEPTCPD. We vary privacy parameter $\epsilon = 0.1, 0.5, 1$ and ∞, representing the non-private case. For each of our simulations, we use $n = 200$ observations where the true change occurs at time $k^* = 100$. This process is repeated 10^4 times. The results of these simulations are presented in 2.1, which plots the empirical probabilities $\beta = \Pr[|\tilde{k} - k^*| > \alpha]$ as a function of α. All the parameters of each simulation are stated in the caption.



(a) Bernoulli: $p_0 = 0.2; p_1 = 0.8$

(b) Bernoulli: $p_0 = 0.2; p_1 = 0.4$

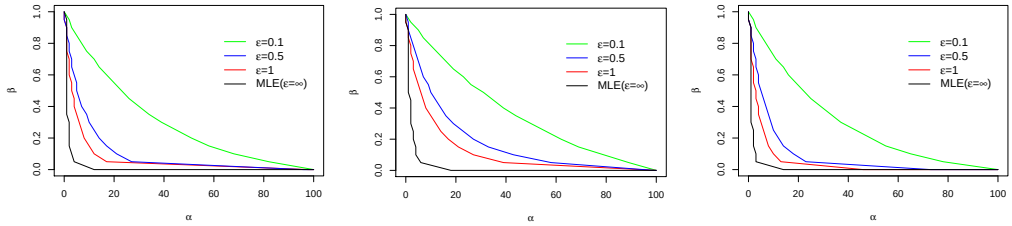
(c) Bernoulli: underspecified p change



(d) Gaussian $\sigma = 1$: $\mu_0 = 0; \mu_1 = 1$

(e) Gaussian $\sigma = 1$: $\mu_0 = 0; \mu_1 = 0.5$

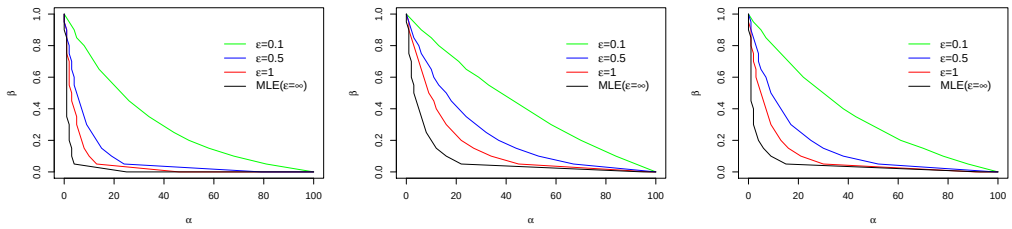
(f) Gaussian: underspecified μ change



(g) Gaussian $\mu = 0$: $\sigma_0 = 1; \sigma_1 = 5$

(h) Gaussian $\mu = 0$: $\sigma_0 = 1; \sigma_1 = 3$

(i) Gaussian: underspecified σ change



(j) Gamma $\theta = 2$: $k_0 = 3; k_1 = 1$

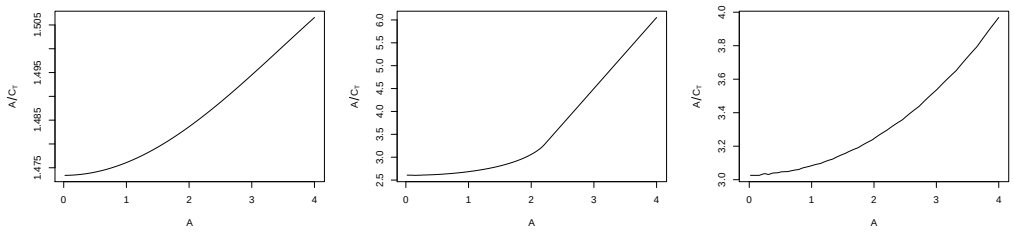
(k) Gamma $\theta = 2$: $k_0 = 3; k_1 = 2$

(l) Gamma: underspecified k change

Figure 2.1: Measured accuracy of offline algorithms on simulated change-point data. For large and small changes (Columns 1 and 2, resp.), parameters specify distributions from which data are drawn and hypothesized distributions given as inputs to the algorithm; for underspecified changes (Column 3), data are drawn according to large change values but algorithm is provided hypothesized distributions consistent with small change values.

2.1 illustrates three important results for our offline algorithms when data are drawn from Bernoulli, Gaussian, or Gamma distributions: accuracy deteriorates as privacy improves but performs quite well even for strong privacy guarantees ($\epsilon < 1$), accuracy is best when the true change in distribution is large (Columns 1 vs 2), and the algorithm performs well even when the true change is larger than that hypothesized (Column 3). The performance in the underspecified change experiments bolster our theoretical results substantially, indicating that our hypotheses can be quite far from the distributions of the true data and our algorithms will still identify a change-point relatively accurately.

Choice of truncation parameter A. The OFFLINEPCPD algorithm does not provide meaningful results when the sensitivity of the log-likelihood ratio is infinite, as in the case of Gaussian and Gamma distributions, so we must instead use OFFLINEPTCPD with some truncation parameter A. Theorem 14 shows that accuracy guarantees are strongest when A/C_A is smallest. Since C_A is a function of the hypothesized distributions as well as A, the value of A should be chosen on a case-by-case basis.



(a) Gaussian large mean change (b) Gaussian large variance change (c) Gamma large shape change

(d) Gaussian large mean change (e) Gaussian large variance change (f) Gamma large shape change

Figure 2.2: First row plots A/C_A as a function of A varying from 0 to 4 for different types of changes; theoretical accuracy bounds are strongest when A/C_A is smallest. Second row shows simulated accuracy under different choices of A for different types of change. Each simulation involves 10^4 runs of OFFLINEPTCPD on data generated by 200 i.i.d. samples from appropriate distributions with change-point $k^* = 100$.

The first row of Figure 2.2 numerically plots A against A/C_A for the large change cases we simulated. The plots suggest that a small A also leads to a small A/C_A, and A/C_A converges to a constant as A goes to 0. The second row verifies optimality of small A by simulation, plotting the empirical probabilities β as a function of accuracy α under different choices of A.

Intuitively, since the mechanism outputs $\operatorname{argmax}_{k\in[n]} \left\{ \sum_{i=k}^{n} \left[\log \frac{P_1(x_i)}{P_0(x_i)} \right]_{-A/2}^{A/2} + \text{Lap}(A/\epsilon) \right\}$, there is a trade-off between how much information is lost from truncation in the first term and how much noise is added in the second term. As $A \to 0^+$, each data point contributes $\pm A/2$. For natural distributions, it appears that giving some data points more weight than others does not provide enough additional information to offset the additional required noise.

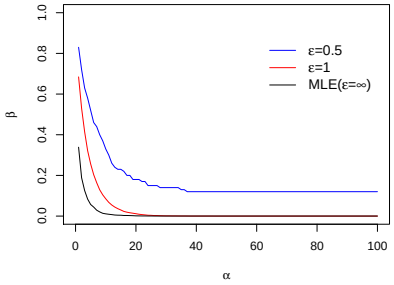
2.5.2 Evaluating the Online Algorithm

We also run Monte Carlo simulations of our online change-point detection algorithm ONLINEPCPD when the data points arrive sequentially and the true change occurs at time $k^* = 5000$. We consider only large mean changes in Bernoulli and Gaussian distributions. For the Gaussian distributions, we truncate the log-likelihoods in the main algorithm and call OFFLINEPTCPD with $A = 0.1$. The new challenge is to choose an appropriate sliding window size n and corresponding threshold T in order to achieve good overall accuracy. The window size of $n = 200$ used in the offline simulations does not permit any threshold that reasonably controls both false positive and false negative rates, so we choose a larger window size of $n = 700$ and restrict our online simulations to $\epsilon = 0.5, 1, \infty$. We choose the appropriate threshold T by setting a constraint that an algorithm must have positive and negative false alarm rates both at most 0.1.

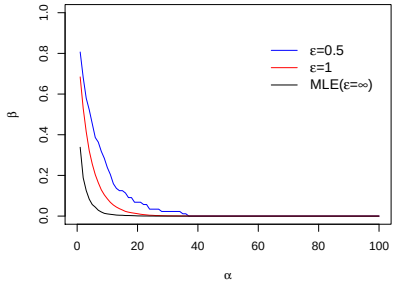
For the online simulations, we chose the lower and upper bounds of T via numerical methods in both Bernoulli and Gaussian models instead of using the theoretical bounds, as these bounds are overly conservative for the Bernoulli model and do not immediately apply for truncation method that is necessary in Gaussian model. We use several key ideas from Section 2.4 to speed up the numerical search of the threshold T. To limit the false positive rate to 0.10 with up to $k^* = 5000$ sliding windows, a conservative lower bound for threshold T is the $1-0.10/5000 = 0.99998$ quantile of the noisy versions of $W_n = \max_{1\le k\le n} \ell(k)$ or $W_n = \max_{1\le k\le n} \ell_A(k)$ with $n = 700$ under the *pre-change* distribution. To limit the false negative rate, an upper bound for threshold T is the 10% quantile of the noisy versions of CUSUM statistics W_n with $n = 700$ when the change occurs at time 350. This will guarantee that the online algorithms raise an alarm with probability at least 0.9 during the time interval $[4650, 5350]$.

To determine these lower and upper bounds for T, we simulate 10^6 realizations of the CUSUM statistics W_{700} in both the pre-change and post-change cases. In each case, we speed up the computation of W_i by using the recursive form $W_i = \max\{W_{i-1}, 0\} +$

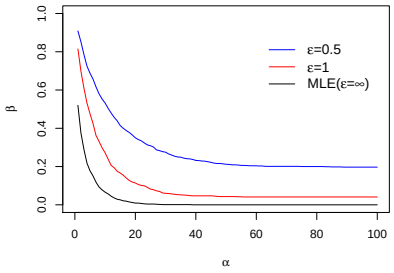
$\log(P_1(X_i)/P_0(X_i))$ or $W_i = \max\{W_{i-1}, 0\} + [\log(P_1(X_i)/P_0(X_i))]_{A/2}^{A/2}$ for $i \geq 1$. The empirical quantiles of the noisy versions of W_{700} under the pre- and post- change cases will yield the lower and upper bounds of the threshold T. When the range of acceptable thresholds T was non-empty, we chose the upper bound. For the Bernoulli model, this resulted in a choice of $T = 220$ for all values of $\epsilon = 0.5, 1, \infty$. In the Gaussian model, we chose $T = 8, 4.5, 100$ for $\epsilon = 0.5, 1, \infty$, respectively. Figure 2.3 (a and c) indeed show that with these parameters, the algorithm works well except with probability about 0.2, and comparison with plots b and d, we can see that almost all of the error for reasonable values of α is due to failure to abort on a window containing the true change-point. This indicates that the primary challenge in the online setting is determining when to raise an alarm in a sequence of sliding windows of observations. Once such window is identified correctly, the offline estimation algorithm can be used to accurately estimate the change-point.



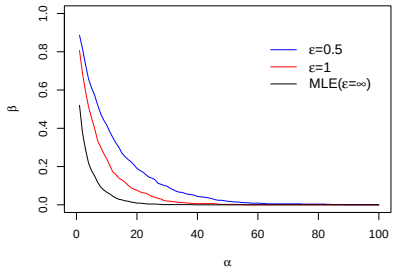
(a) Bernoulli: Online accuracy



(b) Bernoulli: Accuracy when online halts on correct window



(c) Gaussian: Online accuracy



(d) Gaussian: Accuracy when online halts on correct window

Figure 2.3: Probability that the online algorithm produces an inaccurate estimate (left) and probability that the online algorithm produces an inaccurate estimate conditioned on halting in a window containing k^* (right) for Bernoulli and Gaussian large mean changes. Each simulation involves 10^6 runs of ONLINEPCPD or its 0.1-truncated variant with window size $n = 700$ and varying ϵ on data generated by i.i.d. samples from appropriate distributions with change point $k^* = 5000$. See text for description of choices of threshold T.

2.6 Conclusion

This chapter gives private algorithms for both online and offline change-point detection, including the problem of detecting multiple change-points. Our analysis involves providing new finite-sample accuracy guarantees for the standard (non-private) MLE task, and we incorporate tools from differential privacy to add noise to ensure that no individual's data is compromised in the estimation process while maintaining statistical accuracy at a modest privacy cost that depends on the difference between the pre- and post-change distributions.

We extend these results to the online setting by carefully analyzing a range of thresholds for which we can accurately detect a range in a sliding window in which a change-point has likely occurred.

Our empirical results show that change-points in data drawn from well-behaved distributions can be detected relatively accurately even if the hypothesized distributions differ from the real ones. The choice of hypothesized pre- and post-change distributions remains a domain-specific problem, and so rather than provide concrete guidance about how a practitioner should choose these distributions to use our algorithms in a particular setting, this work offers worst-case error bounds on the tradeoff between privacy and accuracy assuming these distributions are chosen correctly, and our algorithms ensure privacy even when they are not. More extensive, domain-specific empirical studies on hypothesis classes of interest is important future work to establish appropriate rules of thumb for practitioners who wish to apply our private change-point tools.

CHAPTER 3

PAPRIKA: PRIVATE ONLINE FALSE DISCOVERY RATE CONTROL

3.1 Introduction

In the modern era of big data, data analyses play an important role in decision-making in healthcare, information technology, and government agencies. The growing availability of large-scale datasets and ease of data analysis, while beneficial to society, has created a severe crisis of reproducibility in science. In 2011, Bayer HealthCare reviewed 67 in-house projects and found that they could replicate fewer than 25 percent, and found that over two-thirds of the projects had major inconsistencies [51]. One major reason is that random noise in the data can often be mistaken for interesting signals, which does not lead to valid and reproducible results. This problem is particularly relevant when testing multiple hypotheses, when there is an increased chance of false discoveries based on noise in the data. For example, an analyst may conduct 250 hypothesis tests and find that 11 are significant at the 5% level. This may be exciting to the researcher who publishes a paper based on these findings, but elementary statistics suggests that (in expectation) 12.5 of those tests should be significant at that level purely by chance, even if the null hypotheses were all true. To avoid such problems, statisticians have developed tools for controlling overall error rates when performing multiple hypothesis tests.

In hypothesis testing, the *null hypothesis* of no interesting scientific discovery (e.g., a drug has no effect), is tested against the alternative hypothesis of a particular scientific theory being true (e.g., a drug has a particular effect). The significance of each test is measured by a p-*value*, which is the probability of the observed data occurring under the null hypothesis, and a hypothesis is *rejected* if the corresponding p-value is below some (fixed) significance level. Each rejection is called a *discovery*, and a rejected hypothesis is

a *false discovery* if the null hypothesis is actually true. When testing multiple hypotheses, the probability of a false discovery increases as more tests are performed. The problem of *false discovery rate (FDR) control* is to find a procedure for testing multiple hypotheses that takes in the p-values of each test, and outputs a set of hypotheses to reject. The goal is to minimize the number of false discoveries, while maintaining high true positive rate (i.e., true discoveries).

In many applications, the dataset may contain sensitive personal information, and the analysis must be conducted in a privacy-preserving way. For example, in genome-wide association studies (GWAS), a large number of single-nucleotide polymorphisms (SNPs) are tested for an association with a disease simultaneously or adaptively. Prior work has shown that the statistical analysis of these datasets can lead to privacy concerns, and it is possible to identify an individual's genotype when only minor allele frequencies are revealed [52]. Therefore, we need formal privacy guarantees for the FDR control problem.

Related Work. The only prior work on differentially private FDR control [53] considers the classic offline multiple testing problem, where an analyst has all the hypotheses and corresponding p-values upfront. Their private method repeatedly applies REPORTNOISYMIN [19] to the celebrated Benjamini-Hochberg (BH) procedure [54] in offline multiple testing to privately pre-screen the p-values, and then applies the BH procedure again to select the significant p-values. The (non-private) BH procedure first sorts all p-values, and then sequentially compares them to an increasing threshold, where all p-values below their (ranked and sequential) threshold are rejected. The REPORTNOISYMIN mechanism privatizes this procedure by repeatedly (and privately) finding the hypothesis with the lowest p-value.

Although the work of [53] showed that it was possible to integrate differential privacy with FDR control in multiple hypothesis testing, the assumption of having all hypotheses and p-values upfront is not reasonable in many practical settings. For example, a hospital may conduct multi-phase clinical trials where more patients join over time, or a market-

ing company may perform A/B testings sequentially. In this chapter, we focus on the more practical *online hypothesis testing problem*, where a stream of hypotheses arrive sequentially, and decisions to reject hypotheses must be made based on current and previous results before the next hypothesis arrives. This sequence of the hypotheses could be independent or adaptively chosen. Due to the fundamental difference between the offline and online FDR procedures, the method of [53] based on REPORTNOISYMIN cannot be applied to the online setting. Instead, we use SPARSEVECTOR, described in Section 1.2, as a starting point. Discussion of non-private online multiple hypothesis testing appears in Section 3.2.1.

Our Results. We develop a differentially private online FDR control procedure for multiple hypothesis testing, which takes a stream of p-values and a target FDR level and privacy parameter ε, and outputs discoveries that can control the FDR at a certain level at any time point. Such a procedure provides unconditional differential privacy guarantees (to ensure that privacy will be protected even in the worst case) and satisfy the theoretical guarantees dictated by the FDR control problem.

Our algorithm, Private Alpha-investing P-value Rejecting Iterative sparse veKtor Algorithm (PAPRIKA, Algorithm 10), is presented in Section 3.3. Its privacy and accuracy guarantees are stated in Theorem 20 and 21, respectively. In Section 3.4, we provide a thorough empirical investigation of PAPRIKA .

3.2 Preliminaries

3.2.1 Background on Online False Discovery Rate Control

In the online false discovery rate (FDR) control problem, a data analyst receives a stream of hypotheses on the database D, or equivalently, a stream of p-values $p_1, p_2, \ldots$. The analyst must pick a threshold α_t at each time t to reject the hypothesis when $p_t \leq \alpha_t$; this threshold can depend on previous hypotheses and discoveries, and rejection must be decided before

the next hypothesis arrives.

The error metric is the false discovery rate, formally defined as: $\text{FDR} = \mathbb{E}[\text{FDP}] = \mathbb{E}\left[\frac{|\mathcal{H}^0 \cap \mathcal{R}|}{|\mathcal{R}|}\right]$, where $\mathcal{H}^0$ is the (unknown to the analyst) set of hypotheses where the null hypothesis is true, and $\mathcal{R}$ is the set of rejected hypotheses. We will also write these terms as a function of time t to indicate their values after the first t hypotheses: $\text{FDR}(t),\ \text{FDP}(t),\ \mathcal{H}^0(t),\ \mathcal{R}(t)$. The goal of FDR control is to guarantee that for any time t, the FDR up to time t is less than a pre-determined quantity $\alpha \in (0, 1)$.

Such a problem was first investigated by [56], who proposed a framework known as *online alpha-investing* that models the hypothesis testing problem as an investment problem. The analyst is endowed with an initial budget, can test hypotheses at a unit cost, and receives an additional reward for each discovery. The alpha-investing procedure ensures that the analyst always maintains an α-fraction of their wealth, and can therefore continue testing future hypotheses indefinitely. Unfortunately, this approach only controls a slightly relaxed version of FDR, known as ***mFDR***, which is given by $\text{mFDR}(t) = \frac{\mathbb{E}\left[|\mathcal{H}^0 \cap \mathcal{R}|\right]}{\mathbb{E}[|\mathcal{R}|]}$. This approach was later extended to a class of generalized alpha-investing (GAI) rules [57]. One subclass of GAI rules, the Level based On Recent Discovery (LORD), was shown to have consistently good performance in practice [58, 59]. GAI++ in [60] improves the class of GAI, with LORD++ as an explicit example. The SAFFRON procedure, proposed by [61], further improves the LORD procedures by adaptively estimating the proportion of true nulls, and is the current state-of-the-art in online FDR control for multiple hypothesis testing.

To understand the main differences between the SAFFRON and the LORD procedures, we first introduce an oracle estimate of the FDP as $\text{FDP}^*(t) = \frac{\sum_{j \le t, j \in \mathcal{H}^0} \alpha_j}{|\mathcal{R}(t)|}$. The numerator $\sum_{j \le t, j \in \mathcal{H}^0} \alpha_j$ overestimates the number of false discoveries, so $\text{FDP}^*(t)$ overestimates the FDP. The oracle estimator $\text{FDP}^*(t)$ cannot be calculated since $\mathcal{H}^0$ is unknown. LORD's naive estimator $\sum_{j \le t} \alpha_j / |\mathcal{R}(t)|$ is a natural overestimate of $\text{FDP}^*(t)$. The SAFFRON's threshold sequence is based on a novel estimate of FDP as $\widehat{\text{FDP}}_{\text{SAFFRON}}(t) =$

$\frac{\sum_{j \le t} \alpha_j \frac{I(p_j > \lambda_j)}{1-\lambda_j}}{|\mathcal{R}(t)|}$, where $\{\lambda_j\}_{j=1}^{\infty}$ is a sequence of user-chosen parameters in the interval $(0, 1)$, which can be a constant or a deterministic function of the information up to time $t-1$. This estimate provides the null-proportion adaptivity basis for SAFFRON.

Our private algorithm is built upon the LORD++ and the SAFFRON algorithms, which are given formally in Algorithm 8 and 9. As a class of GAI, the LORD++ and the SAFFRON both start off with an error budget, which will be allocated to different tests over time. The wealth budget decays as each hypothesis is tested, and it earns back wealth on every rejection except for the first. The decay factors γ_j that depreciate past wealth is a non-increasing sequence summing to one, which ensures that the sum of the wealth budget is always below the desired level α. SAFFRON involves an additional candidacy checking step to be null-proportion adaptive: it never loses wealth when testing candidate p-values with $p_j < \lambda_j$. The sequence $\{\lambda_j\}_{j=1}^{\infty}$ can be defined by any coordinatewise non-decreasing function g_t. For example, $\{\lambda_j\}_{j=1}^{\infty}$ can be a deterministic sequence of constants, or $\lambda_t = \alpha_t$, as in the case of alpha-investing. These λ_j values serve as a weak overestimate of α_j. The algorithm first checks if a p-value is below λ_j, and if so, adds it to the *candidate set* of hypotheses that may be rejected. It then computes the α_j threshold based on current wealth, current size of the candidate set, and the number of rejections so far, and decides to reject the hypothesis if $p_j \le \alpha_j$.

Algorithm 8 SAFFRON($\alpha, W_0, \{\gamma_j\}_{j=0}^{\infty}$)

Input: stream of p-values $\{p_1, p_2, \ldots\}$, target FDR level α, initial wealth $W_0 < \alpha$, positive non-increasing sequence $\{\gamma_j\}_{j=0}^{\infty}$ of summing to one.

Set rejection number $i = 0$

for each p-value p_t **do**

Set $\lambda_t = g_t(R_{1:t-1}, C_{1:t-1})$

Set the indicator for candidacy $C_t = I(p_t < \lambda_t)$. Set the candidates after the j-th rejection as $C_{j+} = \sum_{i=\tau_j+1}^{t-1} C_i$

if $t = 1$ **then**

Set $\alpha_1 = (1 - \lambda_1)\gamma_1 W_0$

else

Compute $\alpha_t = (1 - \lambda_t)(W_0\gamma_{t-C_{0+}} + (\alpha - W_0)\gamma_{t-\tau_1-C_{1+}} + \sum_{j\geq 2} \alpha\gamma_{t-\tau_j-C_{j+}})$

end if

Output $R_t = I(p_t \leq \alpha_t)$

if $R_t = 1$ **then**

Update rejection number $i = i + 1$. Set the i-th rejection time as $\tau_i = t$

end if

end for

Algorithm 9 LORD++$(\alpha, W_0, \{\gamma_j\}_{j=0}^{\infty})$

Input: stream of p-values $\{p_1, p_2, \ldots\}$, target FDR level α, initial wealth $W_0 < \alpha$, positive non-increasing sequence $\{\gamma_j\}_{j=0}^{\infty}$ of summing to one.

Set rejection number $i = 0$

for each p-value p_t **do**

Compute $\alpha_t = W_0\gamma_t + (\alpha - W_0)\gamma_{t-\tau_1} + \sum_{j\geq 2} \alpha\gamma_{t-\tau_j}$

Output $R_t = I(p_t \leq \alpha_t)$

if $R_t = 1$ **then**

Update rejection number $i = i + 1$. Set the i-th rejection time as $\tau_i = t$

end if

end for

Both LORD++ and SAFFRON require that the input sequence of p-values are still valid p-values given past information, which is formalized as *conditional super-uniformity* of null p-values, with respect to a filtration process on the sequence of rejection decisions $\{R_j\}$ and candidacy $\{C_j\}$ (for SAFFRON). It requires that the input sequence of p-values are not too correlated under the null hypothesis. This condition is formalized through a *filtration* on the sequence of candidacy and rejection decisions. Intuitively, this means that the sequence of hypotheses cannot be too adaptively chosen, otherwise the p-values may become overly correlated and violate this condition. Denote by $R_j := I(p_j \leq \alpha_j)$ the indicator for rejection, and let $C_j := I(p_j \leq \lambda_j)$ be the indicator for candidacy. Define the filtration formed by the sequences of σ-fields $\mathcal{F}^t := \sigma(R_1, \ldots, R_t, C_1, \ldots, C_t)$, and let $\alpha_t := f_t(R_1, \ldots, R_{t-1}, C_1, \ldots, C_{t-1})$, where f_t is an arbitrary function of the first $t-1$ indicators for rejections and candidacy. The null p-values are said to be *conditionally super-uniformly distributed* with respect to the filtration $\mathcal{F}$ if:

$$\text{If null hypothesis } H_i \text{ is true, then } \Pr(p_t \leq \alpha_t | \mathcal{F}^{t-1}) \leq \alpha_t. \tag{3.1}$$

We note that independent p-values is a special case of the conditional super-uniformity condition of (3.1). When p-values are independent, they satisfy the following condition:

If the null hypothesis H_i is true, then $\Pr(p_t \leq u) \leq u$ for all $u \in [0, 1]$.

SAFFRON provides the following accuracy guarantees under this condition.

Theorem 19 ([61]). *If the null p-values are conditionally super-uniformly distributed, then we have:*
(a) $\mathbb{E}\left[\sum_{j \leq t, j \in \mathcal{H}^0} \alpha_j \frac{I(p_j > \lambda_j)}{1-\lambda_j}\right] \geq \mathbb{E}\left[|\mathcal{H}^0 \cap \mathcal{R}(t)|\right]$*;*
(b) The condition $\widehat{FDP}_{SAFFRON}(t) \leq \alpha$ *for all* $t \in \mathbb{N}$ *implies that* $mFDR(t) \leq \alpha$ *for all* $t \in \mathbb{N}$.
If the null p-values are independent of each other and of the non-null p-values, and $\{\alpha_t\}$ and $\{\lambda_t\}$ are coordinatewise non-decreasing functions of the vector $R_1, \ldots, R_{t-1}, C_1, \ldots, C_{t-1}$, then
(c) $\mathbb{E}\left[\widehat{FDP}_{SAFFRON}(t)\right] \geq \mathbb{E}\left[FDP(t)\right] := FDR(t)$ *for all* $t \in \mathbb{N}$*;*
(d) The condition $\widehat{FDP}_{SAFFRON}(t) \leq \alpha$ *for all* t *implies that* $FDR(t) \leq \alpha$ *for all* $t \in \mathbb{N}$.

3.2.2 Background on Offline Private False Discovery Rate Control

Differential privacy guarantees are often achieved by adding noise scales with the additive sensitivity. Unlike the conventional use of additive sensitivity, [53] defined the notion of *multiplicative sensitivity* specifically for p-values. It is motivated by the observation that, although the additive sensitivity of a p-value may be large, the relative change of the p-value on two neighboring datasets is stable unless the p-value is very small. This notion allows us to treat the logarithm of the p-values as having additive sensitivity η, substantially reducing the scale of noise required to preserve privacy.

Definition 3 (Multiplicative Sensitivity [53]). *A p-value function p is said to be (η, μ)-multiplicative sensitive if for all neighboring databases D and D', either both $p(D), p(D') \leq$*

μ *or*

$$exp(-\eta)p(D) \leq p(D') \leq \exp(\eta)p(D).$$

3.3 Private online false discovery rate control

In this section, we provide our algorithm for private online false discovery rate control, PAPRIKA, given formally in Algorithm 10. It starts with SAFFRON, using SPARSEVECTOR to ensure privacy of the rejection set. However, the combination of these tools is far from immediate, and several algorithmic innovations are required, including: dynamic thresholds in SPARSEVECTOR to accommodate the alpha-investing rule, adding noise that scales with the multiplicative sensitivity of p-values to reduce the noise required for privacy, shifting the SparseVector threshold to accommodate FDR as a novel accuracy metric, and the candidacy indicator step which cannot be done privately and requires modifications to the wealth updates. We resolve this by using a similar wealth updating rule as in LORD++. We provide new analysis for both privacy and accuracy. We elaborate on the algorithmic details and why these modifications are needed to ensure privacy and FDR control.

The non-private online false discovery rate control algorithms decide to reject hypothesis t if the corresponding p-value p_t is less than the rejection threshold α_t; that is, if $p_t \leq \alpha_t$. We instantiate the SPARSEVECTOR framework in this setting, where p_t plays the role of the t^{th} query answer $f_t(X)$, and α_t plays the role of the threshold. Note that SPARSEVECTOR uses a single fixed threshold for all queries, while our algorithm PAPRIKA allows for a dynamic threshold that depends on the previous output. Our privacy analysis of the algorithm accounts for this change and shows that dynamic thresholds do not affect the privacy guarantees of SPARSEVECTOR. However, the algorithm would not be private if the dynamic thresholds also depend on the data. Note that SAFFRON never loses wealth when testing candidate p-values with $p_j \leq \lambda_j$, and the threshold α_j depends on the data since it is based on current wealth. We remove such dependence in PAPRIKA by losing wealth at every

step regardless of whether we test a candidate p-values, similar to LORD++. This will result in stricter FDR control (and potentially weaker power) because our wealth decays faster.

Algorithm 10 PAPRIKA($\alpha, \lambda, W_0, \gamma, c, \epsilon, \delta, s$)

Input: stream of p-values $\{p_1, p_2, \ldots\}$ with mutiplicative sensitivity (η,μ), target FDR level α, initial wealth $W_0 < \alpha$, positive non-increasing sequence $\{\gamma_j\}_{j=0}^{\infty}$ of summing to one, expected number of rejections c, privacy parameters ϵ, δ, threshold shift magnitude s, maximum number of p-values k.

Let $Z_\alpha^0 \sim \text{Lap}(2\eta c/\epsilon)$, count $= 0$,

$A = \frac{sc\eta}{\epsilon} \log \frac{2}{3\min\{\delta, 1-((1-\delta)/\exp(\epsilon))^{1/k}\}}$

for each p-value p_t **do**

if count $\geq c$ **then** Output $R_t = 0$

else

Sample $Z_t \sim \text{Lap}(4\eta c/\epsilon)$. Set $\lambda_t = g_t(R_{1:t-1}, C_{1:t-1})$. Set the indicator for candidacy $C_t = I(\log p_t < \log 2\lambda_t)$.

if $t = 1$

then Set $\alpha_1 = (1 - 2\lambda_1)\gamma_1 W_0$

else

Compute $\alpha_t = (1 - 2\lambda_t)(W_0\gamma_t + (\alpha - W_0)\gamma_{t-\tau_1} + \sum_{j\geq 2} \alpha\gamma_{t-\tau_j})$

if $C_t = 1$ and $\log p_t + Z_t \leq \log \alpha_t - A + Z_\alpha^{\text{count}}$

then Output $R_t = 1$. Set count = count $+1$ and sample $Z_\alpha^{\text{count}} \sim \text{Lap}(2\eta c/\epsilon)$

else Output $R_t = 0$

end for

Similar to prior work on private offline FDR control [53], we use *multiplicative sensitivity* as described in Definition 3, as p-values may have high sensitivity and require unacceptably large noise to be added to preserve privacy. We assume that our input stream

of p-values $p_1, p_2, \ldots,$ each has multiplicative sensitivity (η, μ). As long as μ is small enough (i.e., less than the rejection threshold), we can treat the logarithm of the p-values as the queries with additive sensitivity η. Because of this change, we must make rejection decisions based on the logarithm of the p-values, so our reject condition is $\log p_t + Z_t \leq \log \alpha_t + Z_\alpha$ for Laplace noise terms Z_t, Z_α drawn from the appropriate distributions.

The accuracy guarantees of SPARSEVECTOR ensure that if a value is reported to be below threshold, then with high probability it will not be more than α_{SV} above the threshold. However, to ensure that our algorithm satisfies the desired bound $FDR \leq \alpha$, we require that reports of "below threshold" truly do correspond to p-values that are below the desired threshold α_t. To accommodate this, we shift our rejection threshold $\log \alpha_t$ down by a parameter A. A is chosen such that the algorithm satisfies (ε, δ)-differential privacy, but the choice can be seen as inspired by the α_{SV}-accuracy term of SPARSEVECTOR as given in Theorem 7. Therefore our final reject condition is $\log p_t + Z_t \leq \log \alpha_t - A + Z_\alpha$. This ensures that "below threshold" reports are below $(\log \alpha_t - A) + \alpha_{SV} \approx \log \alpha_t$ with high probability. Empirically, we see that the bound of A in Theorem 20 may be overly conservative and lead to no hypotheses being rejected, so we allow an additional scaling parameter s that will scale the magnitude of shift by a factor of s. The conservative bounds of Theorem 20 correspond to $s = 4$, but in many scenarios a smaller value of $s = 1$ or 2 will lead to better performance while still satisfying the privacy guarantee. Further guidance choosing this shift parameter is given in Section 3.4.4.

Even with these modifications, a naive combination of SPARSEVECTOR and SAFFRON would still not satisfy differential privacy. This is due to the *candidacy indicator* step of the algorithm. In the SAFFRON algorithm, a pre-processing candidacy step occurs before any rejection decisions. This step checks whether each p-value p_t is smaller than a loose upper bound λ_t on the eventual reject threshold α_t. The algorithm chooses α_t using an α-investing rule that depends on the number of candidate hypotheses seen so far, and ensures that $\alpha_t \leq \lambda_t$, so only hypotheses in this candidate set can be rejected. These λ

values are used to control $\widehat{\text{FDP}}_{\text{SAFFRON}}(t)$, which serves as a conservative overestimate of $\text{FDP}(t)$. (For a discussion of how to choose λ_t, see Lemma 6 or our experimental results in Section 3.4. Reasonable choices would be $\lambda_t = \alpha_t$ or a small constant such as 0.2.)

Without adding noise to the candidacy condition, there may be neighboring databases with p-values p_t, p_t' for some hypothesis such that $\log p_t < \log \lambda_t < \log p_t'$, and hence the hypothesis would have positive probability of being rejected under the first database and zero probability of rejection under the neighbor. This would violate the $(\epsilon, 0)$-differential privacy guarantee intended under SPARSEVECTOR. If we were to privatize the condition for candidacy using, for example, a parallel instantiation of SPARSEVECTOR, then we would have to reuse the same realizations of the noise when computing the rejection threshold α_t to still control FDP, but this would no longer be private.

Since we cannot add noise to the candidacy condition, we weaken it in PAPRIKA to be $\log p_t < \log 2\lambda_t$[1] Then if a hypothesis has different candidacy results under neighboring databases and the multiplicative sensitivity η is small, then the hypothesis is still extremely unlikely to be rejected even under the database for which it was candidate. To see this, consider a pair of neighboring databases that induce p-values where $\log p_t < \log 2\lambda_t < \log p_t'$. Due to the multiplicative sensitivity constraint, we know that $\log p_t \geq \log 2\lambda_t - \eta$. Plugging this into the rejection condition $\log p_t + Z_t \leq \log \alpha_t - A + Z_\alpha$, we see that we would need the difference of the noise terms to satisfy $Z_t - Z_\alpha \leq \log \frac{1}{2} - A + \eta$, which by analysis of the Laplace distribution, will happen with exponentially small probability in n when $\eta = \text{poly}^{-1}(n)$.[2] Our PAPRIKA algorithm is thus (ϵ, δ)-differentially private, and we account for this failure probability in our (exponentially small) δ parameter, as stated in Theorem 20.

[1]We note that although this change is algorithmically equivalent to scaling up the parameter λ_t by a factor of 2, this slack is relevant for certain instantiations of PAPRIKA that set $\lambda_t = \alpha_t$, which we show perform well empirically. (See Section 3.4 for more details.) We write this step as a relaxation of the candidacy condition both for notational consistency with existing non-private alpha-investing-based FDR control methods, such as SAFFRON AI [61], that also choose $\lambda_t = \alpha_t$, and to emphasize that this slack in the candidacy condition is necessary in ensuring differential privacy of the overall algorithm.

[2]Such values of η are typical; see examples in Section 3.4 where $\eta = \frac{1}{\sqrt{n}}$. The shift term A also has dependence on η which contributes to the bound.

One may wonder whether this candidacy step in necessary at all. Since we have removed the dependence of α_t on the size of the candidate set in PAPRIKA, the threshold α_t is no longer null-proportion sensitive. The advantage of being null-proportion adaptive in SAFFRON increases as the proportion of non-nulls increases, but we focus on the case where the non-nulls are sparse, and thus it has little impact in our setting. In Section 3.4, we empirically compare PAPRIKA to two private versions of LORD++, which we call PrivLORD and PrivLORD2. The former combines SPARSEVECTOR and LORD++, with the same threshold shifting as described earlier in this section. The latter adds the candidacy checking step on top of PrivLORD. We see in Section 3.4.3 that both methods provide poor FDR control relative to PAPRIKA, thus providing empirical evidence that the candidacy step in PAPRIKA plays a vital role in FDR control, even if α_t is not null-proportion sensitive. Further details about PrivLORD and PrivLORD2 are deferred to Appendix **??**.

Our PAPRIKA algorithm allows analysts to specify a maximum number of hypotheses tested k and rejections c. We require a bound on the maximum number of hypotheses tested because the accuracy guarantees of SPARSEVECTOR only allows exponentially (in the size of the database) many queries to be answered accurately. Once the total number of rejections reaches c, the algorithm will fail to reject all future hypotheses. We do not halt the algorithm as in SPARSEVECTOR and therefore, PAPRIKA does not have a stopping criterion, and we can safely talk about the FDR control at any fixed time, just like SAFFRON and LORD++.

Our algorithm also controls at each time t, $\widehat{\text{FDP}}_{\text{PAPRIKA}}(t) \leq \frac{\sum_{j \leq t} \alpha_t \frac{I(p_j > 2\lambda_j)}{1-2\lambda_j}}{|\mathcal{R}(t)|}$. We note that this is equivalent to $\widehat{\text{FDP}}_{\text{SAFFRON}}(t)$ by scaling down λ_j by a factor of 2. By analyzing and bounding this expression, we achieve FDR bounds for our PAPRIKA algorithm, as stated in Theorem 21.

Theorem 20. *For any stream of p-values* $\{p_1, p_2, \ldots\}$*,* PAPRIKA *is* (ϵ, δ)*-differentially private.*

As a starting point, our privacy comes from SPARSEVECTOR, but as discussed above,

many crucial modifications are required. To briefly summarize the key considerations, we must handle different thresholds at different times, multiplicative rather than additive sensitivity, a modified notion of the candidate set, and introducing a small delta parameter to account for the new candidate set definition and the shift.

Before proving Theorem 20, we will state and prove the following lemma, which will be useful in the proofs of Theorem 20 and Theorem 21.

Lemma 5. *If $Z_1 \sim$ Lap$(2b)$, $Z_2 \sim$ Lap(b) and $C > 0$ is a constant, we have* $\Pr(Z_1 \geq Z_2 - C) = 1 - \frac{2}{3}\exp(-\frac{C}{2b}) + \frac{1}{6}\exp(-C/b)$.

Proof.

$$\begin{aligned}
\Pr(Z_1 \geq Z_2 - C) &= \int_{-\infty}^{\infty}\int_{x-C}^{\infty} \frac{1}{4b}\exp(-\frac{|y|}{2b})\frac{1}{2b}\exp(-\frac{|x|}{b})dydx \\
&= \int_{\infty}^{C}(1 - \frac{1}{2}\exp(-\frac{|x-C|}{2b}))\frac{1}{2b}\exp(-\frac{|x|}{b})dx \\
&+ \int_{C}^{\infty}\frac{1}{2}\exp(-\frac{|x-C|}{2b})\frac{1}{2b}\exp(-\frac{|x|}{b})dx \\
&= \int_{-\infty}^{C}\frac{1}{2b}\exp(-\frac{|x|}{b})dx - \int_{-\infty}^{0}\frac{1}{4b}\exp(-\frac{|3x-C|}{2b})dx \\
&- \int_{0}^{C}\frac{1}{4b}\exp(-\frac{C+x}{2b})dx + \int_{C}^{\infty}\frac{1}{4b}\exp(-\frac{|3x-C|}{2b})dx \\
&= 1 - \frac{2}{3}\exp(-\frac{C}{2b}) + \frac{1}{6}\exp(-\frac{C}{b})
\end{aligned}$$

□

Now we are ready to prove Theorem 20.

Proof. Fix any two neighboring databases D and D'. Let R denote the random variable representing the output of PAPRIKA$(D, \alpha, \lambda, W_0, \{\gamma_j\}_{j=0}^{\infty}, c, \epsilon, \delta, s)$ and let R' denote the random variable representing the output of PAPRIKA$(D', \alpha, \lambda, W_0, \{\gamma_j\}_{j=0}^{\infty}, c, \epsilon, \delta, s)$. Let k denote the total number of hypotheses. When $\log p_t \geq \log 2\lambda$ and $\log p'_t \geq \log 2\lambda$ for all t, $\Pr(R = \{0, 0, \ldots, 0\}) = 1 = \Pr(R' = \{0, 0, \ldots, 0\})$. When $\log p_t < \log 2\lambda$ and

$\log p'_t < \log 2\lambda$ for all t, privacy follows from the privacy of SPARSEVECTOR with dynamic thresholds. Since the threshold at each time t only depends on the threshold at time $t-1$ and and private rejection $R(t-1)$, by post-processing, the threshold α_t is private. Then by post-processing and the privacy of SPARSEVECTOR , the rejection $R(t)$ is also private. We give the formal probability argument as follows. For any neighboring D, D' and any sequence of hypotheses, we first consider the output up to the first rejection, which is ABOVETHRESH . Consider any output $r \in \{0,1\}^l$. Let $r = \{r_1, r_2, \ldots, r_l\}$, with $r_l = 1$ and $r_1 = \ldots = r_{l-1} = 0$. Let

$$f_i(D, z, \alpha_i) = \Pr(\log p_i(D) + Z_i < \log \alpha_i - A + z)$$
$$g_i(D, z, \alpha_i) = \Pr(\log p_i(D) + Z_i \geq \log \alpha_i - A + z),$$

where $\alpha_1, \ldots, \alpha_t$ is a fixed sequence of thresholds determined by the r. We have

$$\frac{\Pr(R = r|D)}{\Pr(R' = r|D')} \tag{3.2}$$
$$= \frac{\int_{-\infty}^{\infty} \Pr(Z_\alpha = z) \Pr(R_l(D) = r_l | r_{l-1}, \ldots, r_1) \Pr(R_2(D) = r_2 | r_1) \Pr(R_1(D) = r_1) dz}{\int_{-\infty}^{\infty} \Pr(Z_\alpha = z) \Pr(R_l(D') = r_l | r_{l-1}, \ldots, r_1) \Pr(R_2(D') = r_2 | r_1) \Pr(R_1(D') = r_1) dz}$$
$$= \frac{\int_{-\infty}^{\infty} \Pr(Z_\alpha = z) g_l(D, z, \alpha_l) \prod_{i=1}^{l-1} f_i(D, z, \alpha_i) dz}{\int_{-\infty}^{\infty} \Pr(Z_\alpha = z) g_l(D', z, \alpha_l) \prod_{i=1}^{l-1} f_i(D', z, \alpha_i) dz},$$
$$= \frac{\int_{-\infty}^{\infty} \Pr(Z_\alpha = z - \eta) g_l(D, z - \eta, \alpha_l) \prod_{i=1}^{l-1} f_i(D, z - \eta, \alpha_i) dz}{\int_{-\infty}^{\infty} \Pr(Z_\alpha = z) g_l(D', z, \alpha_l) \prod_{i=1}^{l-1} f_i(D', z, \alpha_i) dz}, \tag{3.3}$$
$$\leq \frac{\int_{-\infty}^{\infty} \exp(\epsilon/2c) \Pr(Z_\alpha = z) g_l(D, z - \eta, \alpha_l) \prod_{i=1}^{l-1} f_i(D', z, \alpha_i) dz}{\int_{-\infty}^{\infty} \Pr(Z_\alpha = z) g_l(D', z, \alpha_l) \prod_{i=1}^{l-1} f_i(D', z, \alpha_i) dz}, \tag{3.4}$$
$$\leq \frac{\int_{-\infty}^{\infty} \exp(\epsilon/2c) \Pr(Z_\alpha = z) \exp(\epsilon/2c) g_l(D', z, \alpha_l) \prod_{i=1}^{l-1} f_i(D', z, \alpha_i) dz}{\int_{-\infty}^{\infty} \Pr(Z_\alpha = z) g_l(D', z, \alpha_l) \prod_{i=1}^{l-1} f_i(D', z, \alpha_i) dz}, \tag{3.5}$$
$$= \exp(\epsilon/c). \tag{3.6}$$

Equation (3.3) is from change of integration variable z to $z - \eta$. Inequality (3.4) is because

Z_α follows Lap$(2\eta c/\epsilon)$ and $\log p_i(D) - \eta \leq \log p_i(D')$. Inequality (3.5) is because

$$\begin{aligned} g_l(D, z-\eta, \alpha_l) &= \Pr(\log p_l(D) + Z_l \geq \log \alpha_l - A + z - \eta) \\ &\leq \Pr(\log p_l(D') + \eta + Z_l \geq \log \alpha_l - A + z - \eta) \\ &\leq \Pr(\log p_l(D') + Z_l \geq \log \alpha_l - A + z - 2\eta) \\ &\leq \exp(\epsilon/2c) \Pr(\log p_l(D') + Z_l \geq \log \alpha_l - A + z) \\ &\leq \exp(\epsilon/2c) g_l(D', z, \alpha_l). \end{aligned}$$

When we restart ABOVETHRESH after the first rejection, the inital threshold is the post-processing of the previous ouputs, which is also private. Then by simple composition, the overall privacy loss is ϵ.

For other cases, the worst case is that for all t, $\log p_t < \log 2\lambda$ and $\log p'_t \geq \log 2\lambda$. In this setting, we have

$$\Pr(R' = r) = \begin{cases} 1 & \text{if } r = \{0, 0, \ldots, 0\} \\ 0 & \text{otherwise.} \end{cases}$$

To satisfy (ϵ, δ)-differential privacy, we need to bound the probability of outputting r for database D. We first consider $r = \{0, 0 \ldots, 0\}$. We wish to bound $\Pr(R' = \{0, 0 \ldots, 0\}) \leq \exp(\epsilon) \Pr(R = \{0, 0, \ldots, 0\}) + \delta$ and $\Pr(R = \{0, 0 \ldots, 0\}) \leq \exp(\epsilon) \Pr(R' = \{0, 0, \ldots, 0\}) + \delta$. The latter is trivial since $\exp(\epsilon) \Pr(R' = \{0, 0, \ldots, 0\}) + \delta = \exp(\epsilon) + \delta$, which is greater than 1. It remains to satisfy $\Pr(R' = \{0, 0 \ldots, 0\}) \leq \exp(\epsilon) \Pr(R = \{0, 0, \ldots, 0\}) + \delta$,

which is equivalent to $1-\delta \leq \exp(\epsilon)\Pr(R=\{0,0,\ldots,0\})$. We have

$$\begin{aligned}
\Pr(R=\{0,0\ldots,0\}) &= \Pr(R_1=0)\Pr(R_2=0|R_1=0)\ldots\Pr(R_k=0|R_{k-1}=0) \\
&= \prod_{t=1}^{k}\Pr(\log p_t + Z_t \geq \log\alpha_t - A + Z_\alpha) \\
&> \prod_{t=1}^{k}\Pr(\log 2\lambda - \eta + Z_t \geq \log\alpha_t - A + Z_\alpha) \qquad (3.7) \\
&= \prod_{t=1}^{k}\Pr(Z_t \geq Z_\alpha + \log\alpha_t - \log 2\lambda + \eta - A) \\
&= \prod_{t=1}^{k}(1-\frac{2}{3}\exp(-\frac{\epsilon(A+\log(2\lambda/\alpha_t)-\eta)}{4\eta c}) \\
&+ \frac{1}{6}\exp(-\frac{\epsilon(A+\log(2\lambda/\alpha_t)-\eta)}{2\eta c})) \qquad (3.8) \\
&\geq \left(1-\frac{2}{3}\exp(-\frac{\epsilon(A+\log 2-\eta)}{4\eta c})\right)^k, \qquad (3.9)
\end{aligned}$$

where Inequality (3.7) is because the worst case happens when p_t is η below the candidacy threshold $\log 2\lambda$, Equation (3.8) applies Lemma 5, and Inequality (3.9) follows from the facts that $\alpha_t \leq \lambda$ for all t and that the third term in (3.8) is positive. Setting (3.9) to be larger than $(1-\delta)/\exp(\epsilon)$, we have,

$$\frac{2}{3}\exp\left(-\frac{\epsilon(A+\log 2-\eta)}{4\eta c}\right) \leq 1-\left(\frac{1-\delta}{\exp(\epsilon)}\right)^{\frac{1}{k}}. \qquad (3.10)$$

Next, we consider all other possible outputs r. Define the set $S := \{r \mid \text{there exists a } t \text{ such that } r_t = 1\}$. We wish to bound $\Pr(R \in S) \leq \exp(\epsilon)\Pr(R' \in S)+\delta$ and $\Pr(R' \in S) \leq \exp(\epsilon)\Pr(R \in S)+\delta$. The latter is trivial since $\Pr(R' \in S) = 0$.

It remains to bound $\Pr(R \in S) \leq \delta$. For any t, we have

$$\begin{aligned}
\Pr(R \in S) &\leq \Pr(R_t = 1) \\
&= \Pr(\log p_t + Z_t \leq \log \alpha_t - A + Z_\alpha) \\
&\leq \Pr(\log 2\lambda + Z_t \leq \log \alpha_t - A + Z_\alpha) \qquad (3.11) \\
&= \Pr(Z_t \leq Z_\alpha - (\log(2\lambda/\alpha_t) + A)) \\
&\leq \Pr(Z_t \leq Z_\alpha - (\log 2 + A)) \\
&= \frac{2}{3}\exp\left(-\frac{\epsilon(A + \log 2)}{4\eta c}\right) - \frac{1}{6}\exp\left(-\frac{\epsilon(A + \log 2)}{2\eta c}\right) \qquad (3.12) \\
&\leq \frac{2}{3}\exp\left(-\frac{\epsilon(A + \log 2)}{4\eta c}\right), \qquad (3.13)
\end{aligned}$$

where Inequality (3.11) is because the worst case occurs when $\log p_t = \log 2\lambda$, Equality (3.12) applies Lemma 5, and Inequality (3.13) follows from the facts that $\alpha_t \leq \lambda$ for all t and that the second term in (3.12) is negative. Setting (3.13) to be less than δ, we have,

$$\frac{2}{3}\exp\left(-\frac{\epsilon(A + \log 2)}{4\eta c}\right) \leq \delta. \qquad (3.14)$$

Combining Equations (3.14) and (3.10), we have the condition that $\frac{2}{3}\exp\left(-\frac{\epsilon(A+\log 2-\eta)}{4\eta c}\right) \leq \min\{\delta, 1 - ((1-\delta)/\exp(\epsilon))^{1/k}\}$.

Rearranging this inequality for A gives

$$A \geq \frac{4\eta c}{\epsilon}\left(\log\frac{2}{3\min\{\delta, 1 - ((1-\delta)/\exp(\epsilon))^{1/k}\}} - \log 2 + \eta\right),$$

which is how the shift term A is set in PAPRIKA.

□

Next we describe the theoretical guarantees of FDR control for our private algorithm PAPRIKA which is an analog of Theorem 19. We modify the notation of the conditional super-uniformity assumption of SAFFRON to incorporate the added Laplace noise. The

conditions are otherwise identical. (See (3.1) for comparison.) We note that independent p-values is a special case of conditional super-uniformity, but this requirement more generally allows for a broader class of dependencies among p-values. Let $R_j := I(p_j + Z_j \leq \alpha_j + Z_\alpha)$ be the rejection decisions, and let $C_j := I(p_j \leq 2\lambda_j)$ be the indicators for candidacy. We let $\alpha_t := f_t(R_1, \ldots, R_{t-1}, C_1, \ldots, C_{t-1})$, where f_t is an arbitrary function of the first $t-1$ indicators for rejections and candidacy. Define the filtration formed by the sequences of σ-fields $\mathcal{F}'^t := \sigma(R_1, \ldots, R_t, C_1, \ldots, C_t, Z_1, \ldots, Z_t, Z_\alpha)$. The null p-values are conditionally super-uniformly distributed with respect to the filtration $\mathcal{F}'$ if when the null hypothesis H_i is true, then $\Pr(p_t \leq \alpha_t | \mathcal{F}'^{t-1}) \leq \alpha_t$. We emphasize that this condition is only needed for FDR control, and that our privacy guarantee of Theorem 20 holds for arbitrary streams of p-values, even those which do not satisfy conditional super-uniformity.

Our FDR control guarantees for PAPRIKA mirror those of SAFFRON (Theorem 19). The first two statements apply if p-values are conditionally super-uniform, and the last two statements apply if the p-values are additionally independent under the null.

Theorem 21. *If the null p-values are conditionally super-uniformly distributed, then we have:*

(a) $\mathbb{E}\left[\sum_{j \leq t, j \in \mathcal{H}^0} \alpha_j \frac{I(p_j > 2\lambda_j)}{1-2\lambda_j}\right] + \delta t \geq \mathbb{E}\left[|\mathcal{H}^0 \cap \mathcal{R}(t)|\right]$*;*

(b)The condition $\widehat{FDP}_{\text{PAPRIKA}}(t) \leq \alpha$ *for all* $t \in \mathbb{N}$ *implies that* $mFDR(t) \leq \alpha + \delta t$ *for all* $t \in \mathbb{N}$.

If the null p-values are independent of each other and of the non-null p-values, and $\{\alpha_t\}$ and $\{\lambda_t\}$ are coordinate-wise non-decreasing functions of the vector $R_1, \ldots, R_{t-1}, C_1, \ldots, C_{t-1}$, then

(c) $\mathbb{E}\left[\widehat{FDP}_{\text{PAPRIKA}}(t)\right] + \delta t \geq \mathbb{E}\left[FDP(t)\right] := FDR(t)$ *for all* $t \in \mathbb{N}$*;*

(d) The condition $\widehat{FDP}_{\text{PAPRIKA}}(t) \leq \alpha$ *for all t implies that* $FDR(t) \leq \alpha + \delta t$ *for all* $t \in \mathbb{N}$.

Relative to the non-private guarantees of Theorem 19, the FDR bounds provided by PAPRIKA are weaker by an additive of δt. In most differential privacy applications, δ is typically required to be cryptographically small (i.e., at most negligible in the size of the

database) [19], so this additional term should have a minuscule effect on the FDR.[3] We note that ϵ plays a role in the analysis of Theorem 21, although it does not appear in FDR bounds. Equation (3.22) shows that the additive slack term δt in Theorem 21 is in fact $\min\left\{\delta, 1-((1-\delta)/\exp(\epsilon))^{\frac{1}{k}}\right\}t$, which is upper bounded by δt.

Proof. For any time $t > 0$, before the total number of rejections reaches c we bound the number of false rejections as follows:

$$\begin{aligned}
\mathbb{E}\left[|\mathcal{H}^0 \cap \mathcal{R}(t)|\right] &\leq \sum_{j\leq t, j\in\mathcal{H}^0} \mathbb{E}\left[I(\log p_j + Z_j \leq \log \alpha_j - A + Z_\alpha)\right] && (3.15)\\
&\leq \sum_{j\leq t, j\in\mathcal{H}^0} Pr(\log p_j \leq \log \alpha_j) + Pr(Z_j \leq Z_\alpha - A)\\
&\leq \sum_{j\leq t, j\in\mathcal{H}^0} \mathbb{E}\left[\alpha_j\right] + Pr(Z_j \leq Z_\alpha - A), && (3.16)
\end{aligned}$$

where Inequality (3.15) follows from the rejection rule before the total number of rejections reaches c, and the number of false rejections is always 0 afterwards. Inequality (3.16) follows from the conditional super-uniformity property. We bound each term in (3.16) separately. Using the law of iterated expectations by conditioning on $\mathcal{F}'^{t-1}$, we can bound the first term of (3.16) as follows:

$$\begin{aligned}
\sum_{j\leq t, j\in\mathcal{H}^0} \mathbb{E}\left[\alpha_j\right] \leq& \mathbb{E}\left[\sum_{j\leq t, j\in\mathcal{H}^0} \alpha_j \mathbb{E}\left[\frac{I(p_j > 2\lambda_j)}{1-2\lambda_j}|\mathcal{F}'^{t-1}\right]\right]\\
=& \mathbb{E}\left[\mathbb{E}\left[\sum_{j\leq t, j\in\mathcal{H}^0} \alpha_j \frac{I(p_j > 2\lambda_j)}{1-2\lambda_j}|\mathcal{F}'^{t-1}\right]\right]\\
=& \mathbb{E}\left[\sum_{j\leq t, j\in\mathcal{H}^0} \alpha_j \frac{I(p_j > 2\lambda_j)}{1-2\lambda_j}\right], && (3.17)
\end{aligned}$$

where Equation (3.17) applies the conditional super-uniformity. Since $\widehat{\text{FDP}}_{\text{PAPRIKA}}(t) \leq \alpha$,

[3] Alternatively, δ could be treated like a tunable parameter to balance the tradeoff between privacy and FDR control. If an analyst has an upper bound on the allowable slack in FDR, say 0.01, then she could set $\delta = 0.01/t$ to ensure her desired bound.

we have,

$$\mathbb{E}\left[\sum_{j\le t, j\in\mathcal{H}^0}\alpha_j\frac{I(p_j>2\lambda_j)}{1-2\lambda_j}\right]\le\alpha\mathbb{E}\left[|\mathcal{R}(t)|\right].$$

Next, we bound the second term in (3.16) as follows:

$$\begin{aligned}\sum_{j\le t, j\in\mathcal{H}^0}\Pr(Z_j\le Z_\alpha-A)\le&\frac{2t}{3}\exp\left(-\frac{A\epsilon}{4\eta c}\right)-\frac{t}{6}\exp\left(-\frac{A\epsilon}{2\eta c}\right)\\ \le&t\min\left\{\delta,1-\left(\frac{1-\delta}{\exp(\epsilon)}\right)^{\frac{1}{k}}\right\}.\end{aligned}$$

Combining this inequality with (3.17), we bound mFDR as

$$\begin{aligned}mFDR:=&\frac{\mathbb{E}\left[|\mathcal{H}^0\cap\mathcal{R}(t)|\right]}{\mathbb{E}\left[|\mathcal{R}(t)|\right]}\\ \le&\alpha+\frac{1}{\mathbb{E}\left[|\mathcal{R}(t)|\right]}\sum_{j\le t, j\in\mathcal{H}^0}\Pr(Z_j\le Z_\alpha-A)\\ \le&\alpha+\min\left\{\delta,1-\left(\frac{1-\delta}{\exp(\epsilon)}\right)^{\frac{1}{k}}\right\}t\\ \le&\alpha+\delta t.\end{aligned}$$

If the null p-values are independent of each other and the non-nullls, and $\{\alpha_t\}$ is a coordinate-wise non-decreasing function of the vector $R_1,\ldots,R_{t-1}$, then we have

$$\begin{aligned}FDR(t)=&\mathbb{E}\left[\frac{|\mathcal{H}^0\cap\mathcal{R}(t)|}{|\mathcal{R}(t)|}\right]\\ =&\sum_{j\le t, j\in\mathcal{H}^0}\mathbb{E}\left[\frac{I(\log p_j+Z_j\le\log\alpha_j-A+Z_\alpha)}{|\mathcal{R}(t)|}\right]\\ \le&\sum_{j\le t, j\in\mathcal{H}^0}\mathbb{E}\left[\frac{\min\{\alpha_j\exp(Z_\alpha-Z_j-A),1\}}{|\mathcal{R}(t)|}\right]\qquad(3.18)\\ \le&\sum_{j\le t, j\in\mathcal{H}^0}\mathbb{E}\left[\frac{\alpha_j}{|\mathcal{R}(t)|}\right]+\Pr(Z_j\le Z_\alpha-A),\qquad(3.19)\end{aligned}$$

where Inequality (3.18) applies the law of iterated expectations by conditioning on $\mathcal{F}'^{t-1}$ and Lemma 6. Inequality (3.19) follows by a case analysis: if $Z_j > Z_\alpha - A$, then $\exp(Z_\alpha - Z_j - A) < 1$, and thus $\frac{\min\{\alpha_j \exp(Z_\alpha - Z_j - A), 1\}}{|\mathcal{R}(t)|}$ reduces to $\frac{\alpha_j}{|\mathcal{R}(t)|}$. On the other hand, if $Z_j \leq Z_\alpha - A$, then $\frac{\min\{\alpha_j \exp(Z_\alpha - Z_j - A), 1\}}{|\mathcal{R}(t)|} \leq \frac{1}{|\mathcal{R}(t)|} \leq 1$, allowing us to upper bound the expectation by the probability of this event.

We bound the first term in (3.19) as follows:

$$\begin{aligned}
\sum_{j \leq t, j \in \mathcal{H}^0} \mathbb{E}\left[\frac{\alpha_j}{|\mathcal{R}(t)|}\right] &\leq \sum_{j \leq t, j \in \mathcal{H}^0} \mathbb{E}\left[\frac{\alpha_j I(p_j > 2\lambda_j)}{(1-2\lambda_j)|\mathcal{R}(t)|}\right] && (3.20)\\
&\leq \mathbb{E}\left[\frac{\sum_{j \leq t} \alpha_j I(p_j > 2\lambda_j)}{(1-2\lambda_j)|\mathcal{R}(t)|}\right] \\
&= \mathbb{E}\left[\widehat{\text{FDP}}_{\text{PAPRIKA}}(t)\right] \\
&\leq \alpha, && (3.21)
\end{aligned}$$

where Inequality (3.20) applies Lemma 6.

It remains to bound the second term in (3.19), which we do using Lemma 5 as follows:

$$\begin{aligned}
\sum_{j \leq t, j \in \mathcal{H}^0} \Pr(Z_j \leq Z_\alpha - A) &\leq \sum_{j \leq t} \Pr(Z_j \leq Z_\alpha - A) \\
&= \frac{2t}{3}\exp(-\frac{A\epsilon}{4\eta c}) - \frac{t}{6}\exp(-\frac{A\epsilon}{2\eta c}) \\
&\leq \min\left\{\delta, 1 - \left(\frac{1-\delta}{\exp(\epsilon)}\right)^{\frac{1}{k}}\right\} t. && (3.22)
\end{aligned}$$

Combining (3.21) and (3.22), we reach the conclusion that $\text{FDR}(t) \leq \alpha + \min\{\delta, 1 - ((1-\delta)/\exp(\epsilon))^{1/k}\}t \leq \alpha + \delta t$.

□

The following lemma is a key tool in the proof of Theorem 21. Though it is qualitatively similar to Lemma 2 in [61], it is crucially modified to show an analogous statement holds under the addition of Laplace noise.

Lemma 6. *Assume $p_1, p_2, \ldots$ are all independent and let $h : \{0,1\}^k \to R$ be any coordinate-wise non-decreasing function. Assume f_t and g_t are coordinate-wise non-decreasing functions and that $\alpha_t = f_t(R_{1:t-1}, C_{1:t-1})$ and $\lambda_t = g_t(R_{1:t-1}, C_{1:t-1})$. Then for any $t \leq k$ such that $H_t \in \mathcal{H}^0$, we have $\mathbb{E}\left[\frac{\alpha_t I(p_t > 2\lambda_t)}{(1-2\lambda_t)h(R_{1:k})} | \mathcal{F}'^{t-1}\right] \geq \mathbb{E}\left[\frac{\alpha_t}{h(R_{1:k})} | \mathcal{F}'^{t-1}\right]$ and $\mathbb{E}\left[\frac{\min\{\alpha_t \exp(Z_\alpha - Z_t - A), 1\}}{h(R_{1:k})} | \mathcal{F}'^{t-1}\right] \geq \mathbb{E}\left[\frac{I(\log p_t + Z_t \leq \log \alpha_t + Z_\alpha - A)}{h(R_{1:k})} | \mathcal{F}'^{t-1}\right]$.*

Proof. The proof is similar to the proof of Lemma 2 in [61] with the addition of i.i.d. Laplace noise.

In a high level, we hallucinate a vector of p-values that are same as the original vector of p-values, except for the t-th index. This allows us to apply the conditional uniformity property, since now p_t is independent of the hallucinated rejections. We then connect the original rejections and the hallucinated rejections by the monotonicity of the rejections.

We perform our analysis using a hallucinated process: let $\tilde{p}^t_{1:k}$ be a copy of $p_{1:k}$ that is identical everywhere except for the t-th p-value which is set to be 1. That is,

$$\tilde{p}_i = \begin{cases} 1 & \text{if } i = t \\ p_i & \text{otherwise.} \end{cases}$$

Also let the hallucinated Laplace noises $\tilde{Z}^t_{1:k}$ be an identical copy of $Z_{1:k}$, and let $\tilde{Z}_\alpha$ be an identical copy of Z_α. The t-th value of $\tilde{Z}^t_{1:k}$ can be arbitrary since we have ensure the event $\{\tilde{p}_t > 2\lambda_t\}$, so it will fail to become a candidate and the values of $\tilde{Z}_t$ will not be relevant. We denote $\tilde{C}_{1:k}$ and $\tilde{R}_{1:k}$ as the candidates and rejections made using $\tilde{p}^t_{1:k}$, $\tilde{Z}^t_{1:k}$, and $\tilde{Z}_\alpha$.

By construction, we have $\tilde{R}_{1:t-1} = R_{1:t-1}$. On the event $\{p_t > 2\lambda_t\}$, we have $R_t = \tilde{R}_t = 0$ and $C_t = \tilde{C}_t = 0$ because $\tilde{p}_t = 1$, so both will fail to become candidates, and hence we have $\tilde{R}_{1:k} = R_{1:k}$ and the following equation holds:

$$\frac{\alpha_t I(p_t > 2\lambda_t)}{(1-2\lambda_t)h(R_{1:k})} = \frac{\alpha_t I(p_t > 2\lambda_t)}{(1-2\lambda_t)h(\tilde{R}_{1:k})}.$$

We note that when $p_t \leq 2\lambda_t$, the above equation still holds since both sides will be zero.

Since $\tilde{R}^t_{1:k}$ is independent of p_t, we have

$$\begin{aligned}
\mathbb{E}\left[\frac{\alpha_t I(p_t > 2\lambda_t)}{(1-2\lambda_t)h(R_{1:k})}|\mathcal{F}'^{t-1}\right] &= \mathbb{E}\left[\frac{\alpha_t I(p_t > 2\lambda_t)}{(1-2\lambda_t)h(\tilde{R}_{1:k})}|\mathcal{F}'^{t-1}\right] \\
&\geq \mathbb{E}\left[\frac{\alpha_t}{h(\tilde{R}_{1:k})}|\mathcal{F}'^{t-1}\right] \qquad (3.23) \\
&\geq \mathbb{E}\left[\frac{\alpha_t}{h(R_{1:k})}|\mathcal{F}'^{t-1}\right] \qquad (3.24)
\end{aligned}$$

where Inequality (3.23) is obtained by taking the expectation only with respect to p_t by invoking the conditional super-uniformity property and independence of p_t and $h(\tilde{R}_{1:k})$, and Inequality (3.24) follows from the facts that $R_i \geq \tilde{R}_i$ for all i and that the function h is non-decreasing.

For the second inequality in the lemma statement, we hallucinate a vector of p-values $\bar{p}^t_{1:k}$ that equals $p_{1:k}$ everywhere except for the t-th p-value which is set to be 0. That is,

$$\bar{p}_i = \begin{cases} 0 & \text{if } i = t \\ p_i & \text{otherwise.} \end{cases}$$

Also let the hallucinated Laplace noises $\bar{Z}^t_{1:k}$ be an identical copy of $Z_{1:k}$, and let $\bar{Z}_\alpha$ be an identical copy of Z_α. We denote $\bar{C}_{1:k}$ and $\bar{R}_{1:k}$ as the candidates and rejections made using $\bar{p}^t_{1:k}$ and $\bar{Z}^t_{1:k}$. By construction, we have $\bar{R}_i = R_i$ and $\bar{C}_i = C_i$ for all $i < t$. On the event that $\{\log p_t + Z_t \leq \log \alpha_t + Z_\alpha - A\}$, since $\bar{p}_t = 0$ and we inject the same Laplace noise, we have $\bar{R}_t = R_t = 1$ and $\bar{C}_t = C_t = 1$, and hence also $\bar{R}_{1:k} = R_{1:k}$. Then the following equation holds:

$$\frac{I(\log p_t + Z_t \leq \log \alpha_t + Z_\alpha - A)}{h(R_{1:k})} = \frac{I(\log p_t + Z_t \leq \log \alpha_t + Z_\alpha - A)}{h(\bar{R}_{1:k})}.$$

We note that when $\log p_t + Z_t > \log \alpha_t + Z_\alpha - A$, the above equation still holds since both sides will be zero. Since $\bar{R}_{1:k}$ and Z_t, Z_α are independent of p_t, we can take conditional

expectations to obtain

$$\mathbb{E}\left[\frac{I(\log p_t + Z_t \leq \log \alpha_t + Z_\alpha - A)}{h(R_{1:k})}|\mathcal{F}'^{t-1}\right] = \mathbb{E}\left[\frac{I(\log p_t + Z_t \leq \log \alpha_t + Z_\alpha - A)}{h(\bar{R}_{1:k})}|\mathcal{F}'^{t-1}\right]$$
$$\leq \mathbb{E}\left[\frac{\min\{\alpha_t \exp(Z_\alpha - Z_t - A), 1\})}{h(\bar{R}_{1:k})}|\mathcal{F}'^{t-1}\right] \tag{3.25}$$
$$\leq \mathbb{E}\left[\frac{\min\{\alpha_t \exp(Z_\alpha - Z_t - A), 1\})}{h(R_{1:k})}|\mathcal{F}'^{t-1}\right], \tag{3.26}$$

where Inequality (3.25) follows by taking expectation only with respect to p_t by invoking the conditional uniformity property and the fact that the support of p-values is $[0, 1]$, and Inequality (3.26) follows from the facts that $h(R_{1:k}) \leq h(\bar{R}_{1:k})$ since $R_i \leq \bar{R}_i$ for all i and that the function h is non-decreasing. □

There are no known theoretical bounds on the statistical power of SAFFRON even in the non-private setting. Instead, we validate power empirically through the experimental results in Section 3.4.

3.4 Experiments

We experimentally compare the FDR and the statistical power of variations of the PAPRIKA and SAFFRON procedures, under different sequences of $\{\lambda_j\}$. Following the convention of [61], we define PAPRIKA-Alpha-Investing, or PAPRIKA AI, to be the instantiation of Algorithm 10 with the sequence $\lambda_j = \alpha_j$, where the rejection threshold matches the α-investing rule, and we use PAPRIKA to denote Algorithm 10 instantiated with a sequence of constant of λ_j, which in our experiments is $\lambda_j = 0.2$. We use $\lambda_j = 0.5$ in SAFFRON.[4] We generally observe that, even under moderately stringent privacy restrictions, PAPRIKA and its AI variant perform comparably to the non-private alternatives,

[4]Recall from Section 3 that our λ_j is equivalent to the λ_j in SAFFRON scaling down by a factor of 2.

with PAPRIKA AI typically outperforming PAPRIKA. This suggests that even though setting λ_j as a fixed constant may be easier for implementation, parameter optimization can lead to meaningful performance improvements. We chose the sequence $\{\gamma_j\}$ to be a constant $1/k$ up to time k. Note that the sequence can be decreasing such as of the form $\gamma_j \propto j^{-s}$ in [61], which controls the wealth to be more concentrated around small values of j. See [61] for more discussion on the choice of $\{\gamma_j\}$. In our experiments, we set the target FDR level $\alpha + \delta t = 0.2$, and thus our privacy parameter δ is set to be bounded by $0.2/800 = 2.5 \times 10^{-4}$. The maximum number of rejections $c = 40$. All the results are averaged over 100 runs. We investigate two settings: the observations come Bernoulli distributions in Section 3.4.1, and the observations are generated from truncated exponential distributions in Section 3.4.2. In Section 3.4.3, we compare our algorithm against other private algorithms. In Section 3.4.4, we discuss our choice of the shift parameter A and give guidance on how to choose this parameter in practice.

3.4.1 Testing with Bernoulli Observations

We assume that the database D contains n individuals with k independent features. The ith feature is associated with n i.i.d. Bernoulli variables $\xi_1^i, \ldots, \xi_n^i$, each of which takes the value 1 with probability θ_i, and takes the value 0 otherwise. Let t_i be the sum of the ith features. A p-value for testing null hypothesis $H_0^i : \theta_i \leq 1/2$ against $H_1^i : \theta_i > 1/2$ is given by $p_i(D) = \sum_{k=t_i}^{n} \frac{1}{2^n}\binom{n}{k}$. [53] showed that p_i is (μ, η)-multiplicatively sensitive for $\mu = m^{-1-c}$ and $\eta \asymp \sqrt{\frac{\log n}{n}}$, where $m \leq \text{poly}(n)$ and c is any small positive constant. We choose $\theta_i = 0.5$ with probability $1 - \pi_1$, and $\theta_i = 0.75$ with probability π_1, for varying values of π_1, which represents the expected fraction of non-null hypotheses. We consider relatively small values of π_1 as most practical applications of FDR control (such as GWAS studies) will have only a small fraction of true "discoveries" in the data.

In the following experiments, we sequentially test H_0^i versus H_1^i for $i = 1, \ldots, k$. We use $n = 1000$ as the size of the database D, and $k = 800$ as the number of fea-

tures as well as the number of hypotheses. Our experiments are run under several different shifts A, but due to space constraints, we only report results in the main body with $A = \frac{c\eta}{\epsilon} \log \frac{2}{3\min\{\delta, 1-((1-\delta)/\exp(\epsilon))^{1/k}\}}$ (i.e., when $s = 1$), which still satisfies our privacy guarantee. Further discussion on the choice of A and additional results under other shift parameters s are deferred to Section 3.4.4. The results are summarized in Figure 3.1, which plots the FDR and statistical power against the expected fraction of non-nulls, π_1. In Figure 3.1(a) and (b), we compare our algorithms with privacy parameter $\epsilon = 5$ to the non-private baseline methods of LORD [58, 59], Alpha-investing [57], and SAFFRON and SAFFRON AI from [61]. In Figure 3.1(c,d) and (e,f), we compare the performance of PAPRIKA AI and PAPRIKA, respectively, with varying privacy parameters $\epsilon = 3, 5, 10$. We also list these values in Table A.1 in Appendix A.2.

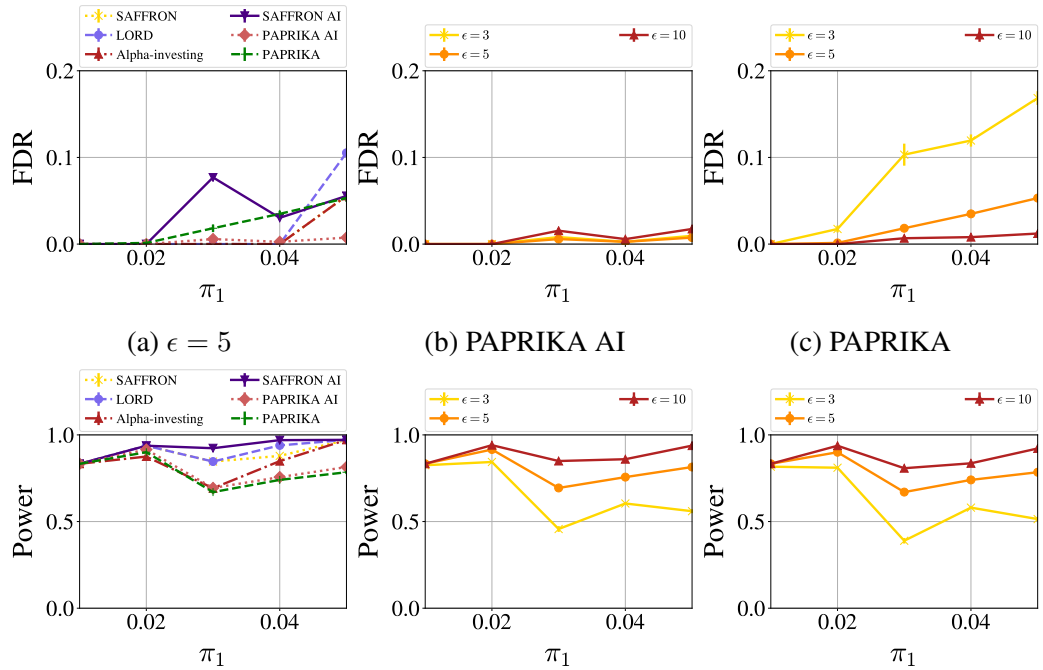


Figure 3.1: FDR and statistical power versus fraction of non-null hypotheses π_1 for PAPRIKA (with $\lambda_j = 0.2$), PAPRIKA AI (with $\lambda_j = \alpha_j$), and non-private algorithms when the database consists of Bernoulli observations.

As expected, the performance of PAPRIKA generally diminishes as ϵ decreases. A notable exception is that FDR also decreases in Figure 3.1(c). This phenomenon is because

we set $\lambda_j = \alpha_j$, resulting in a smaller candidacy set and leading to insufficient rejections. Surprisingly, PAPRIKA AI also yields a lower FDR than many of the non-private algorithms (Figure 3.1(a)), since it tends to make fewer rejections. We also see that PAPRIKA AI performs dramatically better than PAPRIKA, suggesting that the choice of $\lambda_j = \alpha_j$ should be preferred to constant λ_j to ensure good performance in practice.

3.4.2 Testing with Truncated Exponential Observations

In this section, we also assume that have n individuals in the database D, and that each individual's data contains k independent features. The ith feature is associated with n i.i.d. truncated exponential distributed variables $\xi_1^i, \ldots, \xi_n^i$, each of which is sampled according to density

$$f_i(x \mid \theta_i, b) = \frac{\theta_i \exp(-\theta_i x)}{1 - \exp(-b\theta_i)} I(0 \leq x \leq b),$$

for positive parameters b and θ_i. Let t_i be the realized sum of the ith features, and let T_i denote the random variable of the sum of the n truncated exponential distributed variables in the ith entry. A p-value for testing the null hypothesis $H_0^i : \theta_i = 1$ against the alternative hypothesis $H_1^i : \theta_i > 1$ is given by,

$$p_i(D) = \Pr_{\theta_i = 1}(T_i > t_i).$$

[53] showed that p_i is (μ, η)-multiplicatively sensitive for $\mu = m^{-1-c}$ and $\eta \asymp \sqrt{\frac{\log n}{n}}$, where $m \leq \text{poly}(n)$ and c is any small positive constant. In the following experiments, we generate our database using the exponential distribution model truncated at $b = 1$. We set θ_i as follows:

$$\theta_i = \begin{cases} 1 & \text{with probability } 1 - \pi_1 \\ 1.95 & \text{with probability } \pi_1, \end{cases}$$

where we vary the parameter π_1, corresponding to the expected fraction of non-nulls.

We sequentially test H_0^i versus H_1^i for $i = 1, \ldots, k$. We use $n = 1000$ as the size of the database D, and $k = 800$ as the number of features as well as the number of hypotheses. While there is no closed form to compute the p-values, the sum of $n = 1000$ i.i.d. samples is approximately normally distributed by the Central Limit Theorem. The expectation and the variance of ξ_j^i with $b = 1$ are

$$\mathbb{E}\left[\xi_j^i\right] = \frac{1}{\theta_i} + \frac{1}{1 - \exp(\theta_i)}, \text{ and}$$
$$\text{Var}[\xi_j^i] = \frac{1}{\theta_i^2} - \frac{\exp(\theta_i)}{(\exp(\theta_i) - 1)^2},$$

respectively. Therefore, T_i is approximately distributed as $\mathcal{N}(n\mathbb{E}\left[\xi_j^i\right], n\text{Var}[\xi_j^i])$, and we compute the p-values accordingly. We run the experiments with shift $A = \frac{c\eta}{\epsilon} \log \frac{2}{3 \min\{\delta, 1 - ((1-\delta)/\exp(\epsilon))^{1/k}\}}$ (shift magnitude $s = 1$). The results are shown in Figure 3.2, which plots the FDR and statistical power against the expected fraction of non-nulls, π_1.

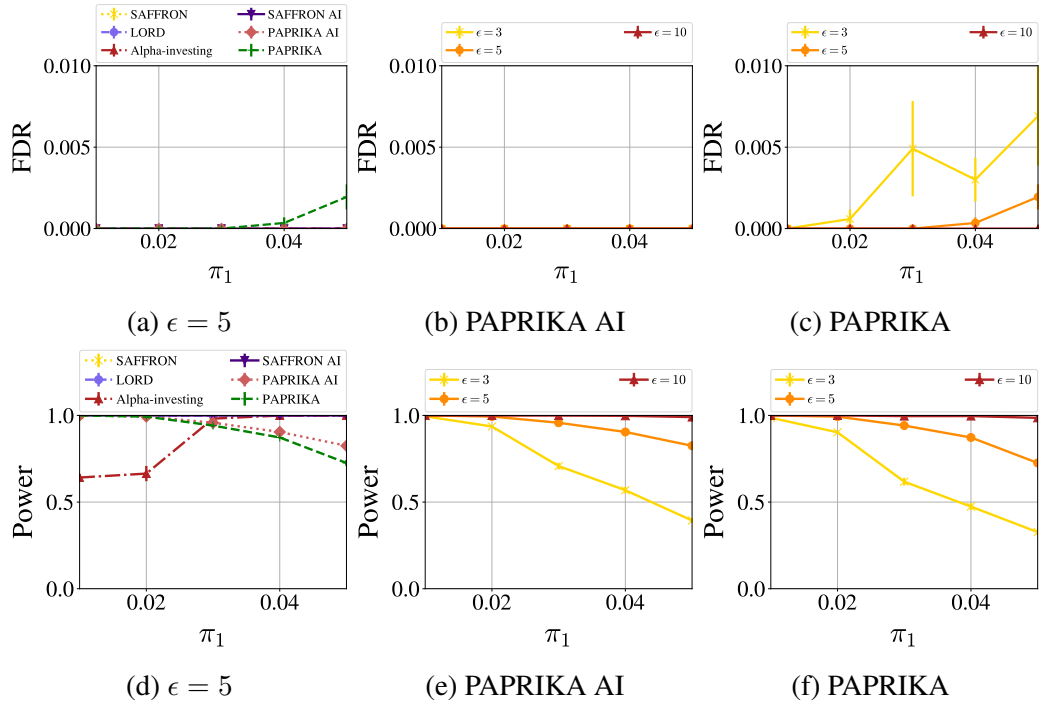


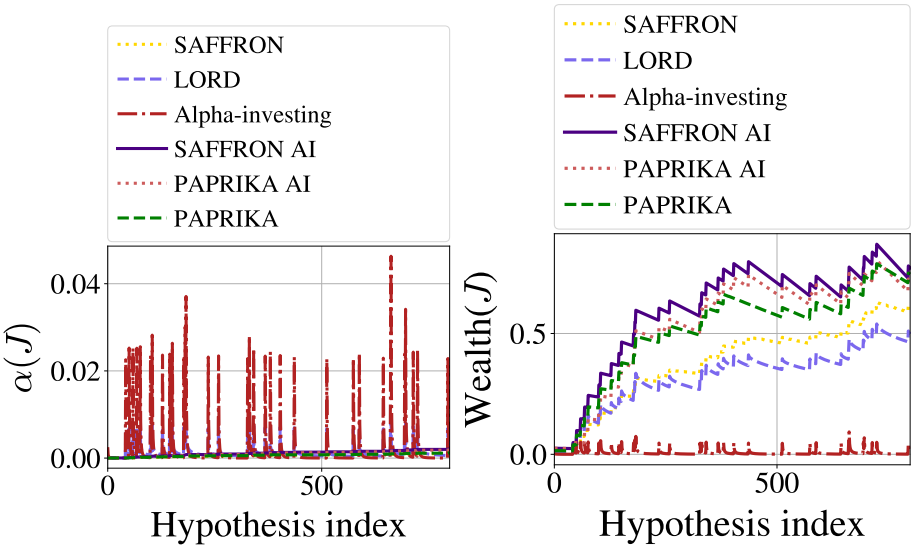
Figure 3.2: FDR and statistical power versus fraction of non-nulls π_1 for PAPRIKA (with $\lambda_j = 0.2$), PAPRIKA AI (with $\lambda_j = \alpha_j$), and non-private algorithms when the database consists of truncated exponential observations.

As in the case with binomial data, we see that the performance of PAPRIKA generally diminishes as ϵ decreases, and that PAPRIKA AI outperforms PAPRIKA, again reinforcing the need for tuning the parameters λ_j based on the alpha-investing rule. All methods perform well in this setting, and the FDR of PAPRIKA AI is visually indistinguishable from 0 at all levels of ϵ and π_1 tested. Numerical values are listed in Table A.2 in Appendix A.2 for ease of comparison.

We provide a further illustration of our experiments on truncated exponentials in Figure 3.3. In particular, we plot the rejection threshold α_t and wealth versus the hypothesis index. Each "jump" of the wealth corresponds to a rejection. We observe that the rejections of our private algorithms are consistent with the rejections of the non-private algorithms, another perspective which empirically confirms their accuracy.

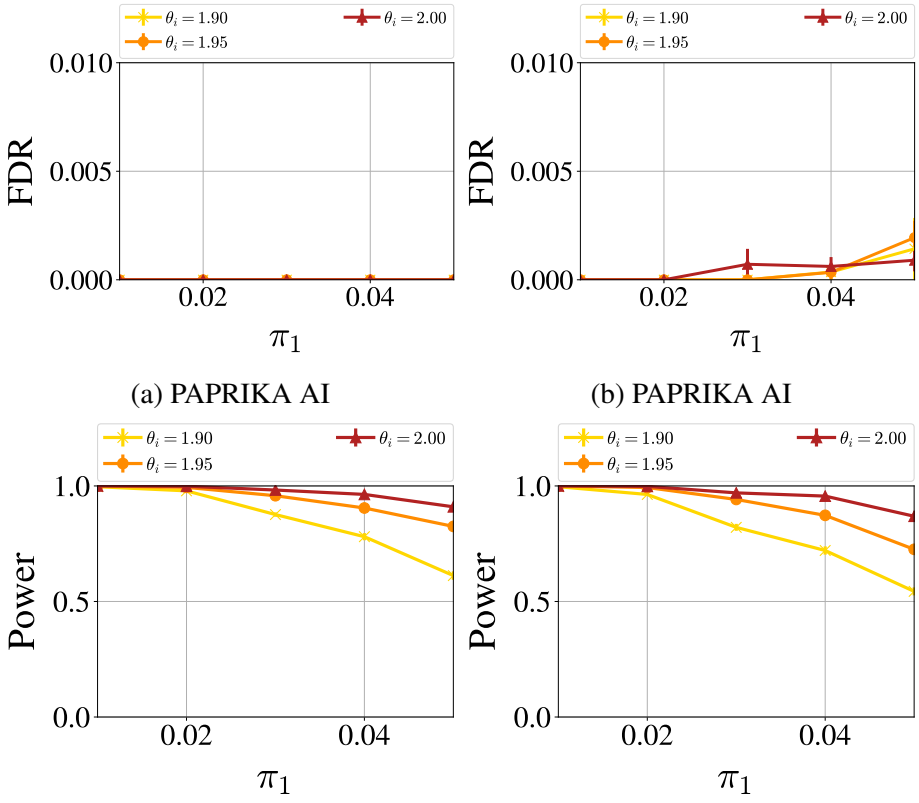
One hypothesis for the good performance observed in Figure 3.2 is that the signal between the null and alternative hypotheses as parameterized by θ_i is very strong, meaning

the algorithms can easily discriminate between the true null and true non-null hypotheses based on the observed p-values. To measure this, we also varied the value of θ_i in the alternative hypotheses. These results are shown in Figure 3.4, which plots FDR and power of PAPRIKA and PAPRIKA AI with when the alternative hypotheses have parameter $\theta_i = 1.90, 1.95, 2.00$. As expected, the performance gets better as we increase the signal, and we observe that when the signal is too weak ($\theta_i = 1.90$), performance begins to decline.



(a) $\epsilon = 5$ (b) $\epsilon = 5$

Figure 3.3: Wealth and rejection threshold α_t versus hypothesis index with privacy parameter $\epsilon = 5$ when the database consists of truncated exponential observations. PAPRIKA AI and SAFFRON AI used $\lambda_j = \alpha_j$, PAPRIKA used $\lambda_j = 0.2$, and SAFFRON used $\lambda_j = 0.5$.



(a) PAPRIKA AI (b) PAPRIKA AI

(c) PAPRIKA (d) PAPRIKA

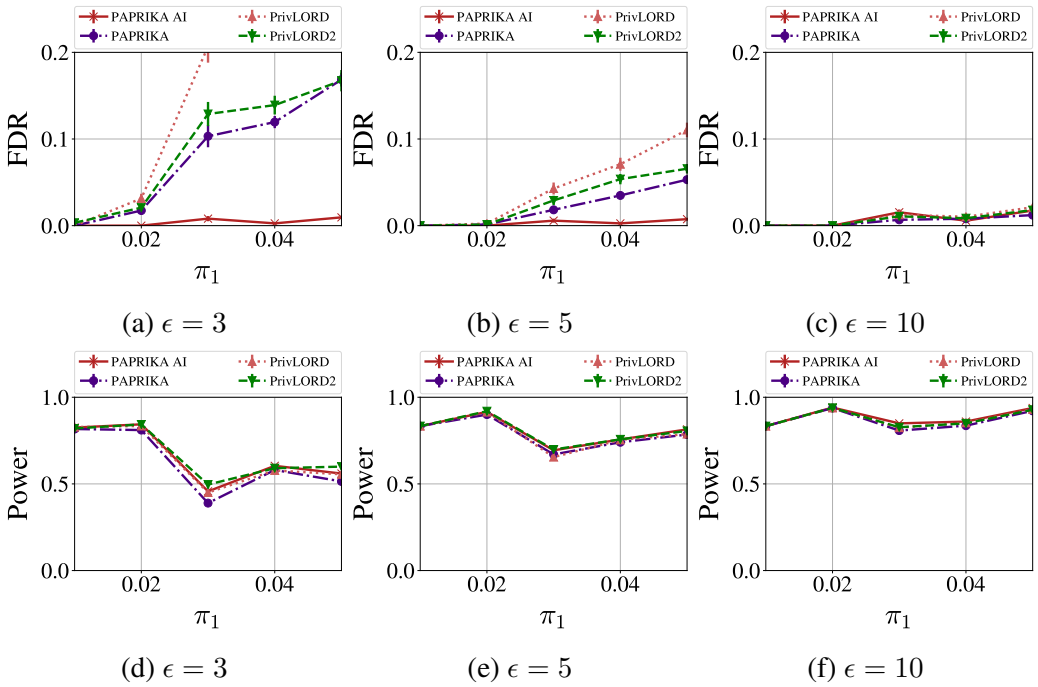
Figure 3.4: FDR and statistical power versus expected fraction of non-null hypotheses π_1 under various choices of signal $\theta_i = 1.90, 1.95, 2.00$ for alternative hypothesis parameters. The privacy parameter is $\epsilon = 5$, and the database consists of truncated exponential observations. The first row shows performance of PAPRIKA AI where $\lambda_j = \alpha_j$, and the second row shows performance of PAPRIKA where $\lambda_j = 0.2$.

3.4.3 Comparison with Other Private Algorithms

As PAPRIKA is the first algorithm for private online FDR control, there is no private baseline for comparison. In Appendix A.2, we show that naïve Laplace privatization of SAFFRON is ineffective. This naïve approach applies the Laplace Mechanism [8] to the p-values of each hypothesis, and then uses these noisy p-values as input to SAFFRON. We see that this baseline mechanism performs extremely poorly relative to PAPRIKA and PAPRIKA AI.

We also compare our PAPRIKA against PrivLORD and PrivLORD2 with Bernoulli observations in Figure 3.5 and truncated exponential observations in Figure 3.6. For com-

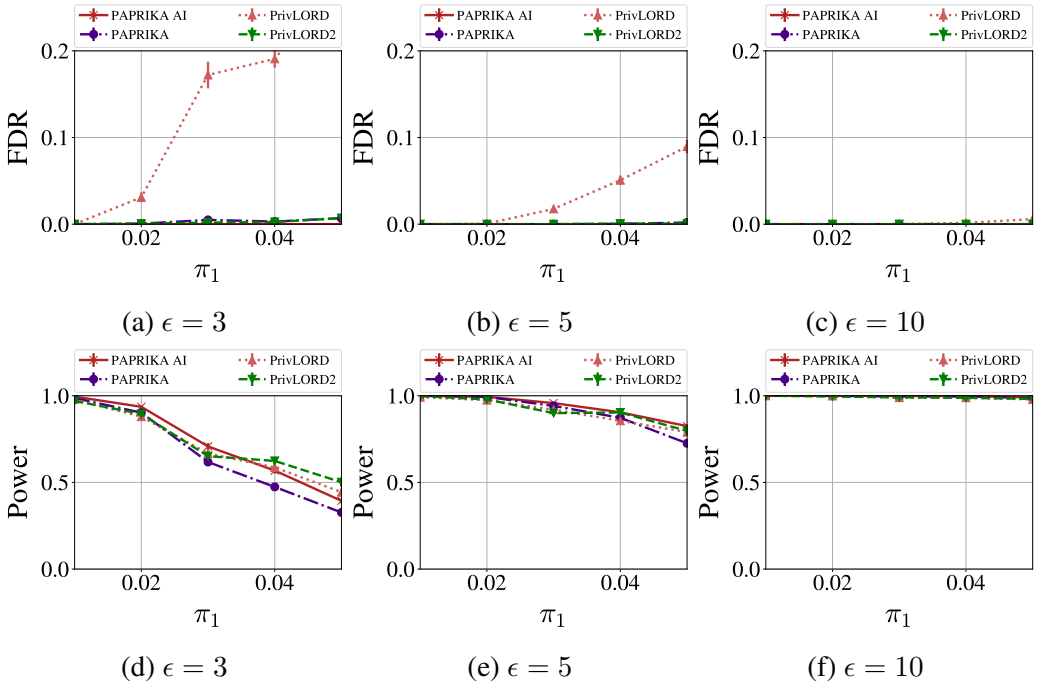
parison, we use the same shift A for the four algorithms, but we note that A should be larger in PrivLORD to control FDR at the level 0.2 as it lacking the candidate checking step, and a larger A leads to worse power, see Section 3.4.4.



(a) $\epsilon = 3$ (b) $\epsilon = 5$ (c) $\epsilon = 10$

(d) $\epsilon = 3$ (e) $\epsilon = 5$ (f) $\epsilon = 10$

Figure 3.5: FDR and statistical power versus fraction of non-nulls π_1 for PAPRIKA (with $\lambda_j = 0.2$), PAPRIKA AI (with $\lambda_j = \alpha_j$), and PrivLORD and PrivLORD2 when the database consists of Bernoulli observations.



(a) $\epsilon = 3$ (b) $\epsilon = 5$ (c) $\epsilon = 10$

(d) $\epsilon = 3$ (e) $\epsilon = 5$ (f) $\epsilon = 10$

Figure 3.6: FDR and statistical power versus fraction of non-nulls π_1 for PAPRIKA (with $\lambda_j = 0.2$), PAPRIKA AI (with $\lambda_j = \alpha_j$), and PrivLORD and PrivLORD2 when the database consists of truncated exponential observations.

We make three key observations. First, PrivLORD makes significantly more false discoveries than the other three algorithms, suggesting that the candidacy checking step largely offsets against the added noise for private algorithms. The performance of PrivLORD gets closer to PAPRIKA and PAPRIKA AI when we add less noise as ϵ goes large. Second, PAPRIKA with constant λ_t has stricter FDR control and slightly weaker power compared to PrivLORD2 as expected, since the threshold α_t in PAPRIKA has an additional constant $(1 - 2\lambda_t)$ factor. Third, PAPRIKA AI provides dramatically better FDR and power tradeoffs—it controls FDR at a much lower level while maintaining power at a similar level as other methods (even the best in Figure 3.5(f) and 3.6(d)), suggesting PAPRIKA with a smart choice of the predictable sequence $\{\lambda_t\}$ is preferred.

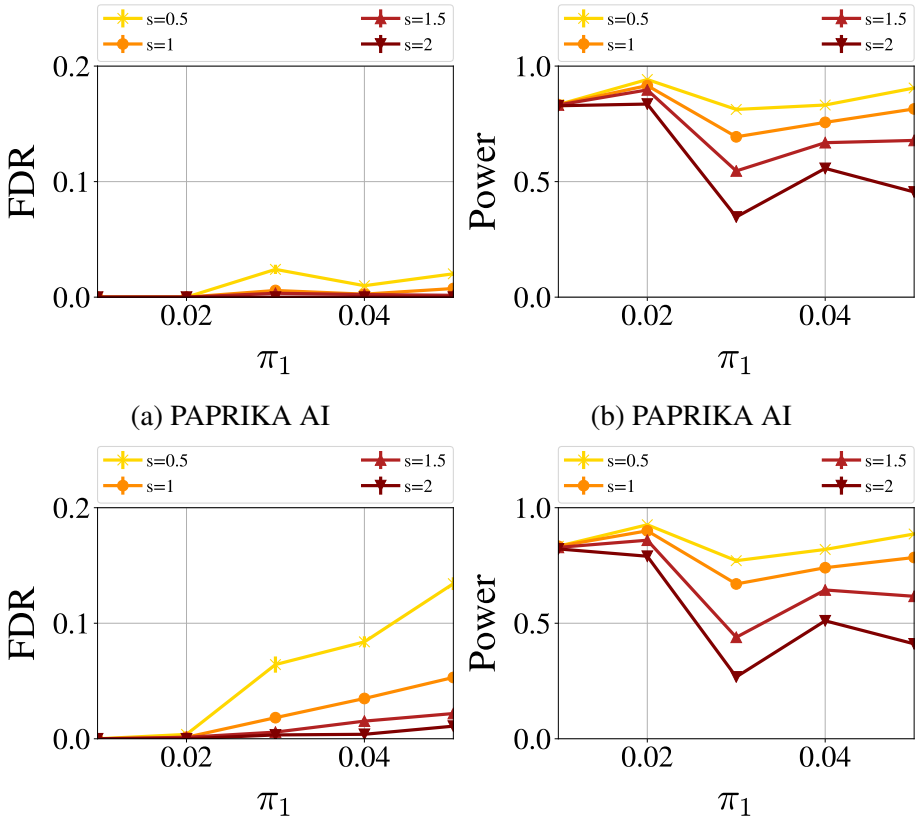
3.4.4 Choice of shift A

We now discuss how to choose the shift parameter A. Theorem 20 gives a theoretical lower bound for A in terms of the privacy parameter δ, but this bound may be overly conservative.

Since the shift A is closely related to the performance of FDR and statistical power, we wish to pick a value of A that yields good performance in practice. In Theorem 21, we show that FDR(t) is less than our desired bound α plus the privacy parameter δt, which naturally requires that the privacy loss parameter δ be small. For a more detailed explanation, we bound Inequality (3.22) in the proof of Theorem 21 using Inequality (3.14) from the proof of Theorem 20, and therefore, the empirical δ is naturally tied to the empirical FDR. As long as we can guarantee the empirical FDR to be bounded by the target FDR level, our privacy loss is bounded by the nominal δ.

We use the Bernoulli example in Section 3.4.1 to investigate the performance under different choices of the shift A with privacy parameter $\epsilon = 5$. The results are summarized in Figure 3.7, which plots the FDR and power versus the expected fraction of non-nulls when we vary the shift size with $s = 0.5, 1, 1.5, 2$.

Larger shifts (corresponding to larger values of s) will lower the rejection threshold, which causes fewer hypotheses to be rejected. This improves FDR of the algorithm, but harms Power, as the threshold may be too low to reject true nulls. Figure 3.7 shows that the shift size parameter s should be chosen by the analyst to balance the tradeoff between FDR and Power, as demanded by the application.



(a) PAPRIKA AI (b) PAPRIKA AI

(c) PAPRIKA (d) PAPRIKA

Figure 3.7: FDR and statistical power versus expected fraction of non-null hypotheses π_1 under various choices of shift magnitude s. The privacy parameter is $\epsilon = 5$, and the database consists of Bernoulli observations. The first row shows performance of PAPRIKA AI where $\lambda_j = \alpha_j$, and the second row shows performance of PAPRIKA where $\lambda_j = 0.2$.

3.5 Conclusion

This chapter gives realizable tools to handle the online private false discovery rate control problem. Our algorithm is based upon two non-private online FDR control algorithms in the GAI family: LORD++ and SAFFRON, and the SPARSEVECTORalgorithm in privacy literature. It involves several algorithmic innovations needed to guarantee privacy and FDR control simultaneously. Our algorithms have strong provable guarantees for privacy and statistical performance as measured by FDR and power. We also provide experimental results to demonstrate the efficacy of our algorithms in a variety of data environments.

CHAPTER 4

DATASET-LEVEL ATTRIBUTE LEAKAGE IN COLLABORATIVE LEARNING

4.1 Introduction

Modern machine learning models have been shown to memorize information about their training data, leading to privacy concerns regarding their use and release in practice. Leakage of sensitive information about the data has been shown via membership attacks [62, 63], attribute inference attacks [64, 65], extraction of text [66] and data used in model updates [67, 68]. These attacks focus on leakage of information about an individual record in the training data, with several recent exceptions [69, 70] pointing out that leakage of global properties about a dataset can also lead to confidentiality and privacy breaches.

In this chaper, we study the problem of leakage of dataset properties at the *population-level*. Attacks on leakage of global properties about the data are concerned with learning information about the *data owner* as opposed to individuals whose privacy may be violated via membership or attribute inference attacks. The global properties of a dataset are confidential when they are related to the proprietary information or IP that the data contains, and its owner is not willing to share. As an example, consider the advantage one can gain from learning demographic information of customers or sales distribution across competitor's products.

Our primary focus is on inferring dataset properties in the *centralized multi-party machine learning setting*. This setting allows multiple parties to increase utility of their data since the model they obtain is trained on a larger data sample than available to them individually. Benefits of computing on combined data have been identified in multiple sectors including drug discovery, health services, manufacturing and finance [71]. For example, anti-money laundering served as a use case for secure data sharing and computation during

the TechSprint organized by the Financial Conduct Authority, UK in 2019 [72]. A potential machine learning task in this setting is to create a system that identifies a suspicious activity based on financial transactions and demographic information of an entity (e.g., a bank customer). Since multiple financial institutions have separate views of the activities, such a system can be used to detect common patterns.

Deployments and availability of secure computation methods [73, 74, 75, 76] can enable multi-party machine-learning by alleviating immediate privacy concerns of the parties. In particular, secure multi-party machine learning provides parties with a *black-box* access to a model trained on their pooled data without requiring the parties to share plaintext data with each other. Unfortunately, as we show in this chapter, this is insufficient to address all privacy implications of collaborative machine learning. In particular, we demonstrate that *global properties* about one party's sensitive attributes can be inferred by the second party, even when only black-box access to the model is available. Consider implications of our attacks in the use case above. An attacker party (e.g., one of the banks) can learn distribution of demographic features pertaining to the customer population in the other bank (e.g., whether the other bank has more female than other customers or what percentage of customers has income over a certain threshold) that it can use in the future when developing a marketing campaign to attract new customers.

Analysis of our attacks shows that leakage of population-level properties is possible even in cases where sensitive attribute is irrelevant to the task, i.e., it has ≈ 0 correlation with the task in hand. Though removing sensitive attributes may seem like a viable solution, it is not provably secure due to correlations that are present in the data. Indeed, we show that in many cases, information is still leaked regardless of whether training data contained the sensitive attribute or not. We argue that this is possible due to correlation between sensitive attributes and other attributes that exists in the data. For example, datasets we use indicate that there is correlation between sets of attributes including gender, occupation and working hours per week, as well as income, occupation and age. Such customer attributes are often

Table 4.1: Comparison of attacks on leakage of dataset properties.

	Attacker's knowledge	**Single-party**	**Multi-party**	**Datasets**
Melis *et al.* [70]	training gradients		✓	tabular, text, images
Ganju *et al.* [69]	model parameters (white-box)	✓		tabular, images
Ateniese *et al.* [77]	model parameters (white-box)	✓		tabular, speech
This work	model predictions (black-box)	✓	✓	tabular, text, graphs

recorded by financial institutions, as a result indicating potential leakage if institutions were to collaborate towards detection of financial crime as described above.

Threat model. We consider the setting where the model is securely trained on the joined data of the honest party and of an *honest-but-curious* party. Honest-but-curious adversary considers a realistic setting where the malicious party (1) will not alter its own data — if it does, the model may not perform well and, if detected, could undermine the trust from the other party in the partnership — and (2) will not change the machine learning code — both parties may wish to observe the code to be run on the data to ensure its quality and security.

The attacker is interested in learning global properties about a sensitive attribute at the dataset level, that is, how values of this attribute are distributed in the other party's dataset. It may be interested in learning which attribute value is dominant (e.g., whether there are more females) or what the precise ratio of attribute values is (e.g., 90% females vs. 70% females).

Attack technique. We show that dataset property can be leaked merely from the black-box access to the model. In particular, the attacker does not require access to the training process of the model (e.g., via gradients [70]) or to model parameters (aka white-box attack [69, 77]). Following other attacks in the space, the attacker also uses shadow models and a meta classifier. However, individual predictions from the model are not sufficient to extract global information about a dataset. To this end, we introduce an attack vector based on a set of queries and use them in combination in order to infer a dataset property. In contrast to previous work on property leakage, the attack requires less information and assumptions on the attacker (see Table 4.1 and Section 4.8 for more details).

Methodology. To understand what causes information leakage about a property we consider several correlation relationships between the sensitive attribute A, the rest of the attributes X, and the target variable Y that the machine learning model aims to learn. Surprisingly, we show that dataset-level properties about A can be leaked in the setting where A has low or no correlation with Y. We demonstrate this with experiments on real data and experiments with a synthetic attribute where we control its influence on X and Y. The attack persists across different model types such as logistic regression, multi-layer perceptrons (MLPs), Long Short Term Memory networks (LSTMs), and Graphical Convolution Networks (GCNs) models and for different dataset types such as tabular, text, and graph data. The attack is efficient as it requires 100 shadow models and fewer than 1000 queries.

Machine learning settings. In addition to the multi-party setting, our property leakage attack can be carried out in the following two settings. (1) single-party setting where an owner of a dataset releases query interface of their model; (2) in the model update setting, one can infer how the distribution of a sensitive property has changed since the previous release of the model. The second attack also applies to multi-party machine learning, showing that the party that joins last exposes its data distribution more than parties who were already collaborating.

Contributions. Our contributions are as follows:

- *Problem Formulation:* We study leakage of properties about a dataset used to train a machine learning model when only black-box access to the model is available to the attacker.

- *Attack Technique:* We propose an effective attack strategy that requires only a few hundred inference queries to the model (black-box access) and relies on a simple attack architecture that even a computationally bound attacker can use.

- *Attack Setting:* We show that leakage of dataset properties is an issue for an owner of a dataset when the owner releases a model trained on their data (single-party setting); when the owner participates in multi-party machine learning, and when the owner contributes data to update an already trained model (e.g., either because it joins other parties or because it has acquired new data).

- *Empirical Results:* We show that distribution of a sensitive attribute can be inferred with high accuracy for several types of datasets (tabular, text, graph) and models, even if the sensitive attribute is dropped from the training dataset and has low correlation with the target variable.

Finally, we note that secure multi-party computation, based on cryptographic techniques or secure hardware, [78, 79, 80, 81, 82, 83, 84, 85, 86, 87] guarantees that nothing except the output of the computation is revealed to the individual parties. However, it is not concerned with what this final output can reveal about the input data of each party. On the other hand, defenses, such as differential privacy, are concerned with individual record privacy and not dataset property privacy considered in this chapter. We discuss this further in Section 4.7. In summary, we believe our work in this chapter identifies a potential gap in multi-party machine learning research in terms of techniques that parties can deploy to protect global properties about their dataset.

4.2 Preliminaries

We assume that there is an underlying data distribution $\mathcal{D}$ determined by variables X, A, Y where X models a set of features, A models a feature that is deemed private (or sensitive) and Y is the target variable, i.e., either a label or a real value (e.g., if using regression models). We consider a supervised setting where the goal is to train a model f such that $f(X, A)$ predicts Y.

Secure multi-party computation (MPC). MPC lets parties obtain a result of a computation on their combined datasets without requiring them to share plaintext data with each other or anyone else. Methods that instantiate it include homomorphic encryption, secret sharing, secure hardware and garbled circuits [88, 89, 90, 91, 92]. These methods vary in terms of their security guarantees (e.g., availability of a trusted processor vs. non-colluding servers) and efficiency. We abstract MPC using an ideal functionality [88]: a trusted third entity accepts inputs from the parties, computes the desired function on the combined data, and returns the output of the computation to each party. Security of protocols implementing this functionality is often captured by proving the existence of a simulator that can simulate adversary's view in the protocol based only on adversary's input and the output of the computation. Hence, an MPC protocol guarantees that an adversarial party learns only the output of the computation but does not learn the content of the inputs of other parties beyond what it can infer based on its own data and the output. Since our attacks are oblivious to the exact technique used for secure computation, we assume ideal MPC functionality and specify additional information available to the adversary in the next section.

Multi-party machine learning. Let D_{honest} and D_{adv} be the datasets corresponding to the data of the victim parties and D_{adv} be the data that belongs to the parties whose data is known to the adversary. For simplicity, we model it using two parties P_{honest} and P_{adv} who own D_{honest} and D_{adv}, respectively. Both D_{honest} and D_{adv} are sampled from $\mathcal{D}$ but may have a different — secret and unknown to the other party — distribution of A, conditional on some latent variable, for example, a party identifier. Parties are interested in increasing the utility of their model through collaboration with each other. To this end, they agree on an algorithm to train a machine learning model, f, using their combined datasets D_{honest} and D_{adv}.

The parties use secure multi-party computation to train f, as they are not willing to share it either due to privacy concerns or regulations. Once the target model is trained, it can be released to the parties either as a white- or black-box. In the former, f is sent to the parties, and, in the latter, the model is available to the parties through an inference interface (e.g., the model stays encrypted at the server such that inferences are made either using secure hardware or cryptographic techniques [93]). We assume that f is trained faithfully and, hence, P_{adv} cannot tamper with how f is trained (e.g., this avoids attacks where a malicious algorithm can encode training data in model weights [94]).

MPC guarantees that parties learn nothing about the computation besides the output, i.e., they learn no other information about each other's data besides what is revealed from their access to f. The goal of this chapter is to show that even by having black-box access to f one party can infer information about other parties' data.

4.3 Data Modeling

To reason about leakage of A's distribution in $\mathcal{D}$, we consider different relationships between X, Y, A based on their correlation. We use $\sim$ to indicate that there is a correlation between random variables and $\perp$ if not. We consider four possible relationships between Y, X and the sensitive attribute A.

$\underline{Y \perp A}$: If Y is independent of A, and if f is faithfully modeling the underlying distribution, A should not be leaked. That is, information about A that an adversary acquires from $f(X, A)$ and $f'(X)$ should be the same for models f and f' trained to predict Y. Two scenarios arise depending on whether the rest of the features are correlated with A or not: $(\boldsymbol{X \perp A, Y \perp A})$ and $(\boldsymbol{X \sim A, Y \perp A})$. We argue that leakage in the latter case is possible due to how machine learning models are trained. Below we describe why it is theoretically feasible and experimentally validate this in Section 4.6.

A machine learning model is trying to learn the conditional probability distribution $\Pr(Y = y|X = x)$ where X are the attributes and Y is the target variable. Suppose there is

a latent variable Z, and the observed X is modeled by $X = h(Z, A)$ where h is a function capturing the relationship between the variables. Even if the target variable Y only depends on Z through a random function g: $Y = g(Z)$, the conditional distribution $\Pr(Y = y|X = x)$ still depends on A. Thus, machine learning models will capture information about A. For example, consider a task of predicting education level (Y) based on data that contains gender (A) and income (X). Suppose income can be modeled by a function of latent variables skill and occupation, and education level is only associated with the skill. Though gender is not correlated with education level ($Y \perp A$), it could be associated with occupation and thus correlated with income (X).

The $(X \sim A, Y \perp A)$ scenario was also noted by Locatello *et al.* [95] when studying fair representations. The authors indicated that even if the original data may not have a bias (i.e., when the target variable and the protected variable are independent) using the protected attribute in training can introduce bias.

To model $(X \sim A, Y \perp A)$ scenario in the experiments, we use correlation coefficients to determine the split of dataset attributes into X and A. To have a more controlled experiment, we also carry out experiments where we introduce a synthetic variable and inject correlations between it and a subset of attributes in X.

$\underline{Y \sim A}$: We also consider two cases where there is a correlation between the target variable Y and the sensitive attribute A: $(\boldsymbol{X \perp A, Y \sim A})$ and $(\boldsymbol{X \sim A, Y \sim A})$. In the setting of $(X \perp A, Y \sim A)$, attribute A and a set of attributes X may be relevant in predicting Y, while being uncorrelated with each other. For example, a reaction of an individual to a new drug (Y) could depend on the age and weight of an adult, while age and weight may be regarded as independent between each other.

The final setting of $(X \sim A, Y \sim A)$ is the most likely scenario to happen in practice where the true distribution and dependence between variables maybe unknown. For example, consider a task of predicting whether a financial transaction by an individual is suspicious or not (Y) based on customer information (e.g., occupation, age, gender) and their

transaction history (X), where their income is the sensitive attribute A. The correlation between attributes could either belong to cases ($\boldsymbol{X \sim A, Y \perp A}$) or to ($X \sim A, Y \sim A$) since attributes such as occupation and age are likely to be correlated with income (as also suggested by the correlations in the datasets we use in our experimental evaluation in Appendix B.1).

4.4 Threat Model and Attack

The goal of the adversarial party P_{adv} is to learn population-level properties about the rest of the dataset used in the multi-party machine learning setting (e.g., in the two-party setting this corresponds to learning properties of the other party's dataset). Since P_{adv} is one of the parties, it has black-box access to the joint model f trained on the data of all the parties (i.e., D_{honest} and D_{adv}). Given this query interface to f, the attacker wants to infer how sensitive attribute A is distributed in honest parties' dataset D_{honest}. Throughout the chapter, we use attribute and feature interchangeably.

We model dataset property leakage as follows. Let $\mathbf{a}_{\mathsf{honest}}$ denote attribute values of A for all records in D_{honest} (for example, if the sensitive attribute is gender, then $\mathbf{a}_{\mathsf{honest}}$ is a vector of gender values of all records in P_{honest} data). We define $p(\mathbf{a}_{\mathsf{honest}})$ to be the property or information about $\mathbf{a}_{\mathsf{honest}}$ that the adversary is trying to infer. For example, the property could be related to determining whether there is a higher presence of female patients in the dataset D_{honest} or learn the exact ratio of female patients.

The attacker, besides knowing its own dataset D_{adv} and having black-box access to the model f, is assumed to have auxiliary dataset D_{aux} that is distributed according to $\mathcal{D}$. Similar to [62], an auxiliary dataset can be generated either via (1) model-based synthesis approach — feeding synthetic data to f and using its output to guide the search towards data samples on which the model returns predictions with high confidence, (2) statistics-based synthesis that uses information about marginal distribution of the attributes, or (3) using a (publicly available) dataset of similar distribution. The attacker can use approach (1) by

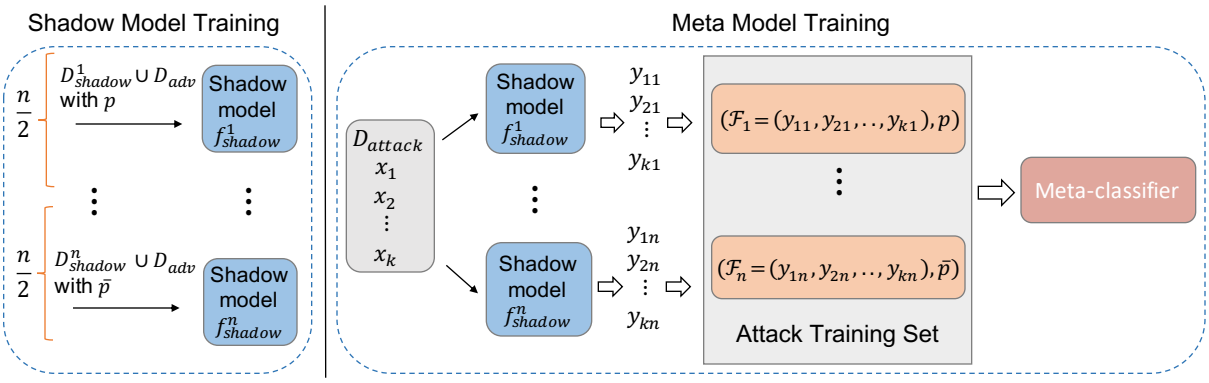


Figure 4.1: Attack model pipeline. Half of shadow models are trained with the property p that the attacker is trying to learn and half without it. Each shadow model f^i_{shadow} is queried on a dataset D_{attack}. Output probability vectors are concatenated to form a vector $\mathcal{F}_i$. Finally, the meta-classifier is trained on feature-label tuples of the form $\{(\mathcal{F}_i, p_i)\}_i$.

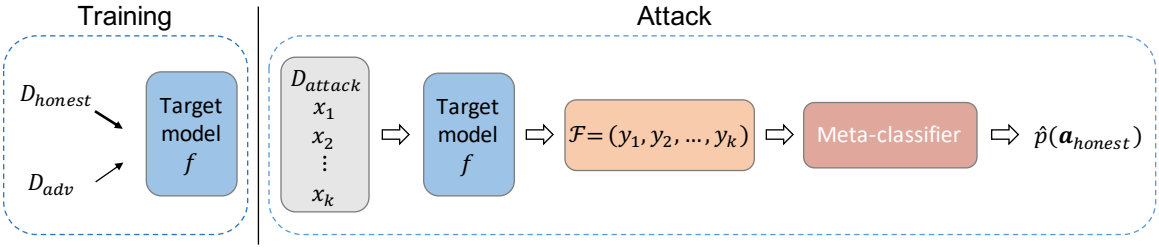


Figure 4.2: Execution of the attack on the target model to learn the prediction of the property $p(\mathbf{a}_{\text{honest}})$ in D_{honest}, $\hat{p}$.

merely using f, while D_{adv} provides it with statistics for (2). The availability of a dataset that follows similar distribution to $\mathcal{D}$ depends on the setting. Consider the anti-money laundering use case in the introduction. A party may have access to billions of financial transactions that it can use either for approach (2) since record-level marginal distribution between demographic features, income, education level is likely to be similar between the parties, or for approach (3) by dividing its dataset into D_{aux} and D_{adv}.

The attack follows the shadow model training approach [77, 62]. However, we modify the attack vector to measure the signal about the distribution of a sensitive attribute in a whole dataset. Our attack strategy is described below; Figure 4.1 shows graphical representation of how the attack model is trained and Figure 4.2 shows the execution of an attack on target model f.

We make an observation that to infer global properties about training data, the attacker

needs to combine information from multiple inferences made by f. To this end, the attacker measures how f performs on a sequence of k records, called D_{attack}, as opposed to a single record used in work on attribute and membership inference. We obtain the "attack feature" sequence $\mathcal{F}$ by setting it to the posterior probability vector across classes returned by f on D_{attack}. Hence, if f is a classification model over l classes $\mathcal{F}$ consists of $k \times l$ values. In the experiments, we construct D_{attack} by sampling from D_{aux} at random. We leave open a question of whether more sophisticated methods of constructing D_{attack} can lead to better attacks.

Shadow models and attack meta-classifier. The attacker relies on shadow models in order to determine whether $\mathcal{F}$ is generated from f trained on a dataset with property p or not. To this end, the attacker trains n "shadow" models that resemble f. In particular, it generates training datasets D^i_{shadow}, half of them exhibiting the property and half not, labeled as p and $\bar{p}$ accordingly. These datasets could be obtained by resampling from D_{aux}. Each shadow model f^i_{shadow} is trained on a dataset $D^i_{\mathsf{shadow}} \cup D_{\mathsf{adv}}$ using the same way as the target central model f. Once f^i_{shadow} is trained, the attacker queries it using D_{attack} and combines inference results to form a feature vector $\mathcal{F}_i$ associated with p or $\bar{p}$, depending on its training data.

After training all shadow models, the adversary has a set of features $\mathcal{F}_i$ with the corresponding property label $p_i \in \{p, \bar{p}\}$. The adversary then trains a meta-classifier on the pairs $\{(\mathcal{F}_i, p_i)\}_i$ using any binary classification algorithm. For example, logistic regression is sufficient for attacks in our experimental evaluation.

The attacker carries out its attack as follows. Once the target model f is trained on the joined data of the attacker and honest party, the attacker queries the model using D_{attack} to obtains the feature representation of the target model, $\mathcal{F}$. It then feeds $\mathcal{F}$ to its meta-classifier and obtains a prediction for the sensitive property $p(\mathbf{a}_{\mathsf{honest}})$.

Single-party attack. We explained the attack strategy for the multi-party case since this is the primary focus of this chapter. However, we can easily adapt the attack to the single-party case: the only change that has to be made to the attack description above is by setting D_{adv} to an empty set. As highlighted in Table 4.1, besides being the first attack on property leakage in the centralized multi-party setting, our attack is also the first to show that dataset properties can be leaked in the black-box setting.

Fine-grained attack. The above attack shows how an adversary can learn whether some property is present in a dataset or not. The attacker can extend this binary property attack and distinguish between multiple properties $P = \{p^1, p^2, \ldots\}$. It simply generates shadow training datasets for each property and then trains a meta-classifier to predict one of the properties in P based on attack vector $\mathcal{F}$. For example, P can be a set of possible ratios of females to other values, and the attack meta-classifier will try to distinguish whether it is 10:90, 50:50 or 90:10 split. In the experimental evaluation, we show that this attack is effective in learning fine-grained distribution of sensitive attributes as well as identifying how the distribution of a sensitive attribute has changed after the model was updated with new data.

Scope. This chapter focuses on understanding the leakage of population-level properties of the training dataset. Since our threat model is similar to that of the attacker who is able to infer individual record-level attributes [64, 94, 65], our setting allows for record-level leakage as well. Albeit, the attack strategy needs to be changed in order to train shadow models that capture the difference between inputs with different attribute values. Importantly, for both the record-level and population-level attribute inference attack, the attacker — here and in [64, 94] — is assumed to know the domain of an attribute it is trying to infer (e.g., `Gender` taking values male, female, or other). Hence, similar to prior work [69, 70], our attack cannot infer a sensitive attribute with a large, potentially unbounded, domain (e.g., such as `Name` for which the attacker may not be able to enumerate all possible values).

4.5 Experimental Setup

The goal of our experiments is to evaluate the efficacy of the attack in Section 4.4 to learn population-level properties about a sensitive attribute in the multi-party and single-party machine learning setting. We then aim to understand how the difference in machine learning models (e.g., logistic regression and neural network models), dataset type (e.g., tabular data, text or graph), access to the model through its weights or inference interface, and attribute correlation influence attack accuracy.

4.5.1 Benchmark Datasets

We evaluate our attack on five datasets described below. The datasets, sensitive attributes, machine learning model tasks, and the type of correlations between the sensitive attribute, other attributes, and the final task are summarized in Table 4.3.

Health [96] The Health dataset (Heritage Health Prize) contains medical records of over $55\ 000$ patients. Similar to the winners of the Kaggle competition, we use 141 features with `MemberID` and `Year` removed. We group the `DaysInHospital` attribute into two classes. The task, Y, is to predict if a patient will be discharged, $\texttt{DaysInHospital} = 0$, or will stay in the hospital, $\texttt{DaysInHospital} > 0$. We consider two sensitive attributes to perform our attack on learning their distribution in the dataset of the benign party: `Gender` and the number of medical claims `ClaimsTruncated`.

Adult [97, 98] The Adult dataset contains US census information including race, gender, income, and education level. The training dataset contains $32\ 561$ records with 14 attributes. We group the education level into four classes: '`Low`', '`Medium-Low`', '`Medium-High`', '`High`'. We use 12 features with `Education` and `Fnlwgt` removed. The task is to predict the class of the `EducationLevel` (i.e., variable Y for this dataset). We again consider two sensitive features whose distribution the attacker is trying to infer: `Gender` and `Income`.

Communities and Crime [98] The Communities and Crime dataset contains $1\,994$ records with 122 features relevant to per capita violent crime rates in the United States, which was also used for evaluating fairness with respect to protected variables [99]. We remove the attributes that have missing data, resulting in 100 attributes. The classification task is to predict the crime rate, i.e., the Y variable is `CrimesPerPop`. We group the crime rate into three classes based on ranges: '< 0.15', '$[0.15, 0.5]$' and '> 0.5', and the task is the multi-class prediction for the crime rate. We consider total percentage of divorce `TotalPctDiv` and `Income` as sensitive features.

Yelp-Health [100] The Yelp dataset contains 5 million reviews of businesses tagged with numerical ratings (1-5) and attributes such as business type and location. We extract a healthcare-related subset that has $2\,384$ reviews for pediatricians and $1\,731$ reviews for ophthalmologists. The classification task is to predict whether the review is positive (rating > 3) or negative (rating ≤ 3). The attack aims to predict the dominant value of the doctor `Specialty` of the benign party.

Amazon [101, 102] The Amazon product co-purchasing network dataset contains product metadata and reviews information about $548\,552$ different products such as books and music CDs. For each product, the following information is available: the similar products that get co-purchased, product type, and product reviews. We use a subset of $20\,000$ products and construct a product co-purchasing network, where each node represents a product and the edge represents if there is at least one reviewer who rated both products, indicating that products are bought by the same user [103]. Each node is associated with one of 4 product types and an average review score from 0 to 5, including half-score reviews (i.e., 11 possible scores in total). The classification task (for a recommendation system) is to predict the average review score of the node given the co-purchasing network and the product types. Depending on the classification task, we split reviewer scores into 2 classes: positive vs. negative review, 6 classes: rounded integer review between 0,1.., 5 and 11

classes: the original review score. The attack aims to predict whether the dominant value of the attribute `ProductType` of the benign party is "books".

4.5.2 Evaluation Methodology

Target model f. We train different target models depending on the dataset type. For tabular data, i.e., Adult, Health, and Crime, we train multinomial logistic regression and fully-connected multi-layer perceptron neural networks (MLP). For the Adult and Crime datasets, we use an MLP network with one hidden layer of size 12 and the last layer with 4 and 3 output classes, respectively. For the Health dataset, we use an MLP network with one hidden layer of size 20 and binary output. In later sections, a neural network model for tabular datasets always refers to an MLP network. In training our target models, we use the Adam [104] optimizer, ReLu as the activation function, a learning rate of 0.01, and a weight decay of 0.0001. For the Yelp-Health dataset, we use the pre-trained glove embedding of dimension 50, a bidirectional LSTM layer of dimension 50. We then use one hidden layer of size 50 and dropout regularization with parameter 0.1 between the last hidden layer and the binary output. For the Amazon dataset, we train the target model using the Graph Convolutional Networks (GCN) [105] with 1 hidden layer of 16 units, Adam as the optimizer, ReLu as the activation function, a learning rate of 0.01, and a weight decay of 0.0005. Each experiment is repeated 100 times, and all attack accuracies are averaged over these runs.

Dataset split. In the multi-party setting, we consider two parties that contribute data for training the target model where one of the parties is trying to learn information about the data of the other party. For Adult and Health datasets, each party contributes 2 000 samples. We use 10 000 or 4 000 samples as D_{aux} to train the shadow models and the attacker uses 1 000 samples in D_{attack} to query the model and obtain the attack vector for the meta-classifier. Table 4.2 summarizes the splits for all other datasets. In Section 4.6.4 we show that a small number of samples in D_{attack} can lead to high attack accuracy as well (e.g., 200

Table 4.2: Dataset split during the attack where $\#D_{\text{attack}}$ is the number of inference queries the attacker makes to the model.

Datasets	$\#D_{\mathsf{adv}}$, $\#D_{\mathsf{honest}}$	$\#D_{\mathsf{aux}}$	$\#D_{\text{attack}}$
Health [96]	2 000	10 000 / 4000	1 000
Adult [97, 98]	2 000	10 000 /4000	1 000
Crime [98]	200	1 500 / 400	94
Yelp-Health [100]	1 000	1 200	200
Amazon [101]	5 000	10 000	1 000

vs. 1 000 for the Amazon dataset).

The distribution of the values of the sensitive attribute A in datasets is determined as follows. We consider the default split of 33:67 in the attacker's data D_{adv} (e.g., 33% of records are books). The attack is evaluated against several D_{honest} datasets for each possible split. For example, we evaluate our attack on 100 D_{honest} datasets: half with 33:67 split and half with 67:33 split in Sections 4.6.1 and 4.6.2. Throughout all experiments, the D_{aux} always has 50:50 split.

Attack setting. We report our main results on attack in the black-box setting; white-box results are deferred to Appendix B.3. We use two different meta-classifiers depending on the target model. For multinomial logistic regression, LSTM and GCN, the meta-classifier model is a binary logistic regression model. For MLP as the target model, we use a two-layer network with 20 and 8 hidden units and a learning rate of 0.001. The meta-classifier models are trained using Adam optimizer.

We perform the attack when the model is trained with the sensitive variable (A) and without it ($\bar{A}$). For the $\bar{A}$ setting, the attribute A is omitted from the machine learning pipeline, including the shadow model training and construction of D_{attack}. This setting allows us to understand the risk of leaking a sensitive attribute, even when that attribute is censored during training. For Yelp-Health, we report only $\bar{A}$ results as LSTM takes the text data, and A would be an additional feature.

Table 4.3: Datasets, tasks and attribute-label correlation where $\sim$ and $\perp$ indicate correlation and no correlation, respectively.

Datasets	Sensitive attribute A	Task Y	Correlation
Health [96]	`Gender` `ClaimsTruncated`	`DaysInHospital`	$X \sim A, Y \perp A$
Adult [97, 98]	`Gender` `Income`	`EducationLevel`	$X \sim A, Y \perp A$ $X \sim A, Y \sim A$
Crime [98]	`TotalPctDivorce` `Income`	`CrimesPerPop`	$X \sim A, Y \sim A$
Yelp-Health [100]	`Specialty`	`ReviewRating`	$X \sim A, Y \perp A$
Amazon [101]	`ProductType`	`ReviewScore`	$X \sim A, Y \sim A$

Types of experiments. We study how correlations between attributes affect the attack. We show that information is leaked even when A is not correlated with the final task. We demonstrate our attack on attribute correlation as present in real dataset distributions (shown in Table 4.3) as well as artificially injected correlation using a synthetic sensitive variable. The latter allows us to control the correlation between the variables.

Real Data. For the experiments where all features are from the real data, including the sensitive variable, we set different variables as sensitive (A) for each dataset and perform a black-box attack using a default split of 33:67 for the sensitive attribute in the attacker's data (D_{adv}).

We compute the pairwise correlation among all the variables using Pearson correlation coefficient [106] for numerical-numerical variables, Cramer's V [107] for categorical-categorical variables, point-biserial correlation coefficient [108] for binary categorical-numerical variables, and ANOVA for multi-level categorical-numerical variables. Based on the observed correlations, for each dataset, we identify the case among those introduced in Section 4.3. Most scenarios correspond to $X \sim A, Y \sim A$. Details on correlation factors for all datasets are deferred to Appendix B.1.

Synthetic Data. For synthetic experiments, we create a new synthetic attribute as our sensitive variable A for the Adult and Health datasets. We add a correlation of A to a subset

of variables in the dataset, denoted as $X' \subseteq X$, and the target variable Y, depending on the cases outlined in Section 4.3. We introduce the correlation by replacing attribute values in X' and/or Y for each record with values that have an injected correlation with A. For Adult dataset, X' is `Income`, for Health dataset, $X' = \{$`DrugCountAve`, `LabCountAve`, `ClaimsTruncated`$\}$. The variable A takes values < 5 or > 5 that are split using 33:67 ratio in the adversarial party's dataset. The honest party has two possible splits: 33:67 ratio and 67:33 ratio. The attacker's goal is to guess the distribution of A in the data of P_{honest}.

4.6 Attack Results

We evaluate for attribute leakage in the following settings: the single-party case where an attacker learns the distribution of an attribute in the training set and the multi-party case where an attacker learns the distribution of an attribute in the data of the honest party. Apart from inferring the dominant attribute (e.g., there are more females than males in a dataset), we perform a fine-grained attack that learns a precise distribution of the two attribute values (e.g., 70% of the dataset are females). We further use this fine-grained attack to infer the change in the attribute distribution in a model update scenario where the model is updated either due to a new party joining or new data arriving.

We report our attack results in the stronger black-box setting for real, synthetic, and fine-grained experiments. We evaluate the white-box attack, where the attacker has access to model parameters, only on the synthetic data. We summarize our key findings below:

- Leakage of sensitive dataset properties in honest party's data is possible even when the sensitive attribute itself is *dropped* during training and has low or no correlation with the final task. We show that the attack accuracy drops only by a few percent when A is not present in many cases.

- An adversary can learn the attribute properties of the honest party's data irrespective of whether it contributes data (multi-party) or not (single-party) to the training

Table 4.4: Multi-Party Setting: Black-box attack accuracy for predicting the value of the distribution of sensitive variable A in the dataset of P_{honest}. The attacker tries to guess whether values of A are split as 33:67 or 67:33 in D_{honest} when its own data D_{adv} has 33:67 split. Columns A and $\bar{A}$ report the accuracy when the sensitive variable is used for training and not, respectively. X' indicates with which features in the dataset and with how many of them A is correlated. Since attack accuracy based on a random guess is 0.5, the attacker is always successful in determining the correct distribution.

Datasets (Output Classes)	Model Type	Attack Accuracy		A	# X'
		A	$\bar{A}$		
Health (2)	Multi-layer Perceptron	.61	.59	`Gender`	24/139
		.75	.71	`ClaimsTruncated`	54/139
Adult (4)	Logistic Regression	.83	.81	`Gender`	5/11
		.98	.96	`Income`	9/11
Crime (3)	Multi-layer Perceptron	.61	.59	`TotalPctDivorce`	26/98
		.78	.60	`Income`	38/98
Yelp-Health (2)	LSTM	-	.74	`Specialty`	review text
Amazon (2)	GCN	.86	.72	`ProductType`	graph
Amazon (6)	GCN	.62	.63	`ProductType`	graph
Amazon (11)	GCN	.67	.61	`ProductType`	graph

dataset.

- For the models and datasets considered in this chapter, our property leakage attack is dataset and model-agnostic and works on tabular, text, or graph data.
- Fine-grained attacks can be used to predict a precise distribution of the attribute as well as learn the change in data distribution during model updates.

4.6.1 Multi-Party Setting

Real Data. Table 4.4 shows the attack accuracy for correlations observed in the real distribution of datasets, with the larger size of D_{aux} as listed in Table 4.1. The attack accuracy with the smaller size of D_{aux} is deferred to Table B.3 in Appendix B.2. We see that the attack accuracy is always better than a random guess in all experiments, regardless of whether the sensitive attribute is included in the training data or not.

We make the following observations. The attack accuracy for Adult data with `Income` as the sensitive attribute is the highest with 98% and 96% when the target model is trained with and without A, respectively. Overall, the attack accuracy ranges between 61-98% when trained with sensitive variable (A) and 59-96% without ($\bar{A}$), respectively. The results for $\bar{A}$ are always lower than with A but are, however, above the random guess baseline of 50%. For the Amazon dataset, we observe that attack accuracy is higher for fewer output classes. We confirm this observation later in Figure 4.4. We also note that the attack accuracy decreases as the size of D_{aux} decreases as shown in Appendix B.2.

To understand how the correlation between A and other features influences the attack, we determine which attributes $X' \subseteq X$ are correlated with A. We set X' to variables based on their correlation factors. Details on how X' of each dataset was determined based on correlation factors is deferred to Appendix B.1. In Table 4.4, # X' denotes the number of attributes correlated with the sensitive attribute A. We note that simultaneously controlling the number of correlated attributes and their correlation strength is hard on real data, so we also use synthetic datasets. We observe that, for the same dataset, the attack accuracy increases with a higher number of correlated attributes X' and the sensitive attribute A.

We show the accuracies for both the pooled model and the honest party's local model in Table B.4 in Appendix B.2. Across all these experiments, we observe a utility increase ranging from 0.58% and 5.90% for the honest party, which motivates the honest party to collaborate and train a joint target model with the other party.

Synthetic Data. Table 4.5 shows our results with a synthetic variable A introduced in the Adult and Health dataset for the multi-party setting. Here, we train the same dataset using both logistic regression and the neural network model (MLP). Recall that the synthetic attribute is introduced to imitate a sensitive variable to control its correlation with other variables. To this end, we create datasets for different correlation criteria among the sensitive variable A, the output Y, and the remaining variables X. We report two findings.

First, logistic regression models appear to be at a higher risk, with average attack accuracy being higher as compared to neural network models: 84.5% vs. 71.3% for Adult and 80.2% vs. 70.8% for Health datasets. We suspect that this is mainly due to their simple architecture, which is easy to learn using a meta-classifier.

Second, the attack works well (greater than 74%) when the sensitive variable A is correlated with the target variable Y irrespective of its relation with X, i.e., cases where $Y \sim A$. The attack accuracy is almost equal to a random guess when $Y \perp A$. Recall that in the case of $X \sim A$, not all features used for training are correlated with A but only those in a subset of X, X'. To understand this scenario further, we reduced the number of features used during training to 3 (we refer to this setting as `R` in the tables). As the number of training features decreases, the correlation signal between A and X' becomes stronger, and the logistic regression model can capture that.

Our experiments for the case when both X and Y are independent of the sensitive variable A exhibit attack accuracy that is close to a random guess. This is expected as the variable has no correlation that the model can memorize, and hence we exclude them from Table 4.5.

4.6.2 Single-Party Setting

In addition to our motivating scenario of the multi-party setting, we evaluate the efficacy of our attack in the single-party setting where the attacker does not contribute towards the training data. For example, this corresponds to a scenario where a model is trained on data from only one hospital and is offered as an inference service for other hospitals. Table 4.6 shows the result for our attack using synthetic data for the Adult and Health dataset when the model is trained using both logistic regression and neural networks. We see that the attack in the single party setting is stronger since the adversary does not provide its own data, which may dilute the signal from the other party. For the case where $Y \sim A$, the attack accuracy is higher than 90%, even if the attribute itself is not used during training.

Table 4.5: Multi-party setting: Black-box attack accuracy for predicting whether the values of (sensitive) synthetic variable A in the data of the honest party are predominantly <5 or >5. The attack accuracy is evaluated on 100 D_{honest} datasets: half with 33:67 and half with 67:33 split. A synthetic correlation with A is added to the variables X and Y depending on the specific case. R corresponds to the setting where only 3 attributes are used for training instead of all data. Attack accuracy based on a random guess is 0.5.

Model	Logistic Regression				Neural Network			
Datasets	Adult		Health		Adult		Health	
Synthetic Variable	A	$\bar{A}$	A	$\bar{A}$	A	$\bar{A}$	A	$\bar{A}$
$X \sim A, Y \sim A$	1.00	1.00	1.00	1.00	.90	.84	.79	.95
$X \perp A, Y \sim A$	1.00	1.00	.99	1.00	.98	.98	.74	.98
$X \sim A, Y \perp A$	.65	.57	.52	.41	.52	.52	.52	.51
$X \sim A, Y \perp A$ (R)	.79	.75	.78	.72	.51	.45	.54	.63

Table 4.6: Single-party setting: Black-box attack accuracy with synthetic data.

Model	Logistic Regression				Neural Network			
Datasets	Adult		Health		Adult		Health	
Synthetic Variable	A	$\bar{A}$	A	$\bar{A}$	A	$\bar{A}$	A	$\bar{A}$
$X \sim A, Y \sim A$	1.00	1.00	.98	1.00	.98	.99	.92	.95
$X \perp A, Y \sim A$	1.00	1.00	.98	1.00	.99	1.00	.89	.98
$X \sim A, Y \perp A$	.67	.60	.48	.53	.56	.52	.52	.49
$X \sim A, Y \perp A$ (R)	.86	.74	.61	.62	.68	.66	.54	.61

This shows that our attack is highly successful even when the attacker does not participate in the training process.

4.6.3 Fine-grained Attack

Information leaked about attribute values can be either in terms of a binary signal, i.e., which attribute value is dominant in the dataset or an exact distribution. The results above show the leakage of the former. To learn information about the exact distribution, we present a variation of our main attack called the fine-grained attack. For this attack, we train a 5-class meta-classifier model that outputs whether a particular value of the sensitive attribute appears in 10%, 30%, 50%, 70%, or 90% of the dataset. Note that we train only one meta-classifier model with 5 output classes, but the attacker can perform a more

Table 4.7: Fine-grained attack accuracy for predicting the precise distribution of sensitive variable A in D_{honest} in the synthetic setting $X \perp A, Y \sim A$, and real data setting when A is `Gender` on the Adult dataset. Attack accuracy based on a random guess is 0.2.

Distribution of A in D_{honest}:	LR Synthetic A		NN Synthetic A		LR A: `Gender`
	A	$\bar{A}$	A	$\bar{A}$	$\bar{A}$
10 : 90	.994	.998	.84	.89	.44
30 : 70	.993	.991	.79	.79	.59
50 : 50	.999	.997	.79	.73	.50
70 : 30	.997	.989	.73	.71	.46
90 : 10	.993	.998	.72	.77	.53

systematic binary search over the distribution by training multiple meta-classifier models. We apply this attack in two settings.

Leakage of Attribute Distribution. We evaluate on the Adult dataset using a synthetic variable A as well as the gender variable. Table 4.7 shows the results for our fine-grained attack for predicting the precise distribution of the sensitive variable. The row $30 : 70$ corresponds to the setting where 30% of records in D_{honest} have the value of the sensitive attribute A less than 5. Here, the attacker tries to guess the split of $30 : 70$ among five possible splits of $10 : 90$, $30 : 70$, etc. The baseline accuracy is 20% because the attacker wishes to distinguish between 5 splits. Since the attack accuracy is always higher than the random guess, the attacker can successfully find the correct ratio by training a meta-classifier that distinguishes between different splits of the sensitive attribute values. Similar to the observation in Section 4.6.1, we observe that logistic regression has higher attack accuracy than neural networks. The attack accuracy for the real data with `gender` as the sensitive attribute is consistently greater than the 20% baseline for random guessing for all the distributions.

Model Update Setting. We apply the fine-grained attack to learn the change in the distribution of an attribute value given access to an updated version of a model. In this attack, the malicious party initially obtains a model that is jointly trained on D_{honest1} and D_{adv}.

Table 4.8: Model update setting: attack accuracy for predicting the dominant value of sensitive variable A in D_{honest2} in the synthetic setting $X \perp A, Y \sim A$ and real data setting when A is `Gender` on Adult dataset when A is removed from the training data. D_{adv} has 50:50 split. Attack accuracy based on a random guess is 0.5.

Distribution of A in D_{honest1}:	Distribution of A in D_{honest2}:	LR Synthetic A	LR A: `Gender`
30:70	30:70	1.00	.87
	70:30	.99	.72
70:30	30:70	.99	.63
	70:30	1.00	.85

Later, another honest party D_{honest2} joins, and a new model is trained on the three parties' data. The attacker tries to infer the dominant value of the sensitive attribute of P_{honest2} given the original and the updated model. It uses a fine-grained attack against both models, as result learning a dominant value in D_{honest1} and $D_{\mathsf{honest1}} \cup D_{\mathsf{honest2}}$. It then compares the two and infers how D_{honest2} has affected the distribution. If the split is dominated by the same attribute value in both models, the attacker uses this attribute value distribution as its guess. Otherwise, the attacker makes a guess that the other attribute value is dominated in D_{honest2}. Table 4.8 shows the results for our attack in the model update setting using synthetic data for the Adult dataset. We observe the attack accuracy is almost close to 100% for the synthetic case and ranges from 63% to 86% for the `Gender` variable. The attack accuracy is high compared to a random guess of 50%.

4.6.4 Attack Parameters

We perform ablation experiments to understand the effect of varying the number of queries, distribution of the sensitive attribute and the number of output classes on the attack accuracy. We use the Amazon graph data for these experiments where, as before, `ProductType` is the sensitive attribute, and `ReviewScore` is the target.

Number of queries. We compute the attack accuracy for two different splits of values of the sensitive attribute, 0:100 (all books) and 30:70 (70% books, 30% of other products),

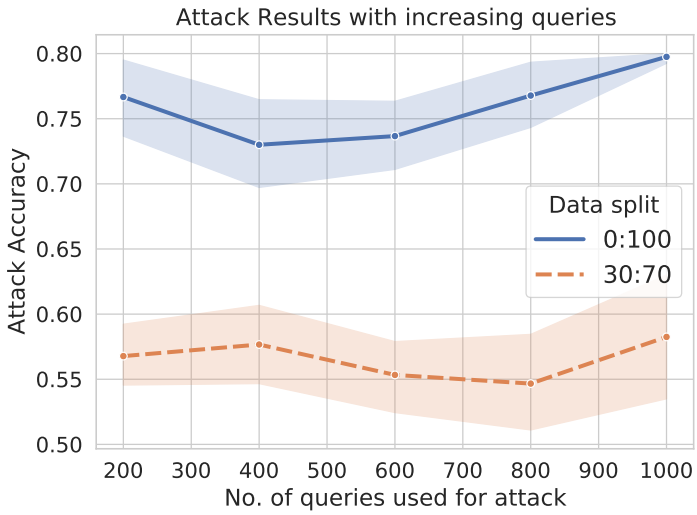


Figure 4.3: Attack accuracy for leaking sensitive attribute `ProductType` on the Amazon graph data (11 output classes) as the number of queries to the model increases.

and train the model to predict one of 11 review scores averaged over 10 runs. Figure 4.3 shows the effect of increasing the number of queries on the attack accuracy. Note that the number of queries also correspond to the input features of our attacker classifier. We observe that changing queries does not significantly impact the attack accuracy. With 1000 queries, attack accuracy is up to 80% for the 0:100 split and ≈59% for 30:70 split.

Attribute distribution and number of output classes. Figure 4.4 shows the results for the GCN trained on the Amazon dataset for 2, 6 and 11 output classes for the review score. We evaluate for all the splits between 0:100 to 100:0. First, we observe that the attack accuracy drops as the ratio of the sensitive attribute values changes from 0:100 to 50:50 and increases again gradually from 50:50 to 100:0. This is because our primary attack is designed to identify the dominant attribute value. For inferring the distribution in the balanced range, the attacker can perform our fine-grained attack discussed in Section 4.6.3. Next, we observe that the attack accuracy is lower for a higher number of output classes such as 6 and 11 as compared to 2. This could be due to lower number of input features that are given to the attack classifier when there are lower number of output classes — the classifier is able to learn the attribute distribution better when the information is divided

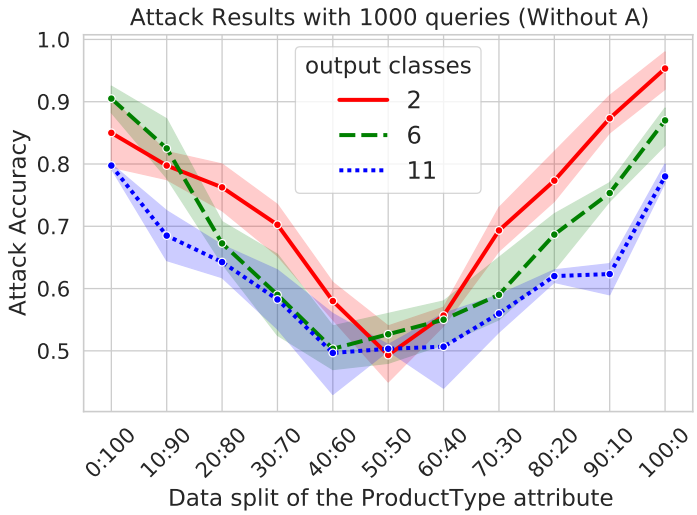


Figure 4.4: Attack accuracy for the Amazon graph data when the sensitive attribute `ProductType` is not used during training for different numbers of output classes across different distributions (splits).

among fewer features thus resulting in a lower dimension input. Similar trends are observed in Figure B.1 in Appendix when A is used during training.

4.7 Defenses

In the previous section, we saw that removing the sensitive attribute from the dataset is not an effective solution due to the correlations that exist between the attributes. Disentangling data representation through variational-auto-encoders [109, 110, 99] allows one to obtain mutually independent variables for representing the data. Intuitively, the removal of this variable before decoding the record for further down-stream tasks would lead to better censorship. Similarly, adversarial learning has also been proposed for learning a privacy-preserving data filter in a multi-party setting [111] and a privacy-preserving record representation [112]. Unfortunately, such techniques do not have provable worst-case guarantees and have been shown ineffective in the privacy context [64].

Differential privacy [113] guarantees record-level privacy, that is, whether a particular record is in their dataset or not. However, differential privacy does not protect population-level properties of a dataset [113, 114]. In fact, a differentially private algorithm with high

utility aims to learn population properties without sacrificing individual privacy. Group differential privacy is an extension of differential privacy that considers the privacy of a group of k correlated records, as a result one way of achieving it is to increase, for example, Laplace noise, proportional to k. Though it can be applied to preserve the privacy of all records in each party's dataset by setting k to the size of each party's data, depending on the setting, it can effect utility as even with $k = 1$ accuracy of models have been shown to drop [115, 116].

In settings with more than two parties, where the attacker controls only one party, the signal weakens as it is harder for the adversary to identify the mapping between a property and a party whose data exhibits it. This was also noted by Melis *et al.* [70] in the federated learning setting with a small number of parties.

4.8 Related work

Membership attacks on machine learning models aim to determine whether a certain record was part of a training dataset or not [62, 63]. These attacks train shadow models that are similar to the target model and use their output (e.g., posterior probabilities over all classes) to build a meta-classifier that classifies records as members of the training data or not based on inference results of the target model on the record in question. A recent link stealing attack on graphs can be seen as a type of a membership attack that tries to infer whether two nodes have a link between them in the training graph [117].

Attribute inference attacks [64, 65], on the other hand, aim to determine the value of a sensitive attribute for a single record. For example, the authors of [64] study leakage of a sensitive value from a latent representation of a record in the model (i.e., a feature extractor layer); an attacker can obtain such intermediate record representations from having access to model parameters. They show that an attribute of a record, even if censored using adversarial learning, can be leaked. Hitaj *et al* [118] show that a malicious party can construct class representatives from a model trained in federated learning setting.

The work by Ganju *et al.* [69] and Ateniese *et al.* [77] are closest to ours as they also consider leakage of *dataset properties*. Different from this chapter, their attack is set in a single-party setting and requires a *white-box access* to the model, i.e., its parameters, that may not always be possible (e.g., when the model access is via cloud-hosted interface). Since the number of model parameters in neural networks can be very large (several million), approaches that are based on sophisticated methods for reducing network representation are required [69]. We show that attacks based on a combination of inferences and logistic regression as a meta-classifier are sufficient to learn attribute distribution.

Property leakage in a multi-party learning has been demonstrated only in federated setting [70]. In this setting an attacker obtains a gradient computed on a small batch of records (e.g., 32) and tries to learn how a sensitive feature is distributed in the batch. This setting is arguably easier from the attacker point of view: an attacker gains access to a much more granular computation on the data compared to the access to a query interface of the final model trained on the whole dataset, as considered in this chapter. Moreover, previous work on dataset property leakage [69, 70, 77] did not consider the case when the sensitive attribute is *removed* from the data and the effect it has on the success of their attacks.

Recently, Zanella-Béguelin *et al.* [68] have demonstrated leakage of text and general trends in the data used to update next word prediction model. Salem *et al.* [67], on the other hand, consider granular leakage about records used to update the model: record labels and their features. Similar to our work, Salem *et al.*use a probing dataset to query the models to obtain the posterior difference. This output is then given to an encoder-decoder framework to reconstruct the meaning of the difference between posteriors of the initial and updated models. Our model update attack, in comparison, is about identifying the distribution of a sensitive feature in the dataset used to update the model and requires a simple machine learning architecture.

4.9 Conclusion

We demonstrate an attack, set in the centralized multi-party machine learning, that lets one of the parties learn sensitive properties about other parties' data. The attack requires only black-box access to the model and can extract the distribution of a sensitive attribute with small number of inference queries. We show that trivial defenses such as excluding a sensitive attribute from training are insufficient to prevent leakage. Our attack works on models for tabular, text, and graph data and datasets that exhibit various correlation relationships among attributes and class labels. Finally, we note that existing techniques for secure computation and differential privacy are either not directly applicable to protect leakage of population-level properties or do so at a high cost.

CHAPTER 5

ATTRIBUTE PRIVACY: FRAMEWORK AND MECHANISMS

5.1 Introduction

Privacy in the computer science literature has generally been defined at the *individual level*, such as differential privacy [8], which protects the value of an individual's data within analysis of a larger dataset. However, there are many settings where confidential information contained in the data goes beyond presence or absence of an individual in the data and instead relates to attributes at the *dataset level*. Global properties about attributes revealed from data analysis may leak trade secrets, intellectual property and other valuable information pertaining to the data owner.

In this chapter, we are interested in privacy of *attributes in a dataset*, where an analyst must prevent global properties of sensitive attributes in her dataset from leaking during analysis. For example, insurance quotes generated by a machine-learned model might leak information about how many female and male drivers are insured by the company that trained the model; voice and facial recognition models may leak the distribution of race and gender among users in the training dataset [77, 119]. Under certain circumstances, even releasing the distribution from which the data were sampled may be sensitive. For example, experimental findings by a pharmaceutical company measuring the efficacy of a new drug would be considered proprietary information.

The naive solution of removing sensitive attributes from the dataset is insufficient, as attributes are often correlated, and protected information can still be leaked by releasing non-sensitive information about the data. Machine learning models have been shown to learn sensitive attributes even when censorship is applied or when the attributes are deemed irrelevant for the actual learning task [64, 70, 120, 69]. In the algorithmic fairness literature,

differential privacy has been used to protect sensitive attributes at the individual level by, e.g., constructing classifiers where an individual's label is differentially private with respect to her race [121]. However, the study of attribute privacy at the dataset or distribution level is limited, both in terms of a framework for reasoning about it and mechanisms for protecting it.

5.1.1 Our Contributions

Problem formulation We initiate the study of *attribute privacy* at the dataset and distribution level and establish the first formal framework for reasoning about these privacy notions. We identify two cases where information about global properties of a dataset may need to be protected: (1) properties of a specific dataset and (2) parameters of the underlying distribution from which dataset is sampled. We refer to the first setting as *dataset attribute privacy*, where the data owner wishes to protect properties of her sample from a distribution, but is not concerned about revealing the distribution. For example, even though the overall prevalence of a disease may be known, a hospital may wish to protect the fraction of its patients with that disease. We refer to the second setting as *distributional attribute privacy*, which considers the distribution parameter itself a secret. For example, demographic information of the population targeted by a company may reveal information about its proprietary marketing strategy. These two definitions distinguish between protecting a sample and protecting the distribution from which the dataset is sampled.

Definitions of Attribute Privacy. We propose definitions for capturing dataset and distributional attribute privacy by instantiating a general privacy framework called the Pufferfish framework [18]. This framework was originally introduced to handle correlations across individual entries in a database. Instantiating this framework for attribute privacy is non-trivial as it requires reasoning about secrets and parameters at a dataset level.

For dataset attribute privacy, our definition considers the setting where individual records

are independent of each other while correlations may exist between attribute values of each record. Then, to be able to capture general global properties of a dataset that need to be protected, we choose to express secrets as functions over attribute values across all records in a dataset. For example, this allows one to express that the average income of individuals in a dataset being below or above \$50K is secret information.

Our second definition also instantiates the Pufferfish framework while explicitly capturing the random variables used to generate attribute values of a record. Here, the parameters of the distribution of protected attributes are treated as confidential information. For example, in a dataset where records capture trials in a stochastic chemical environment, one can express that determining whether the probability with which a certain compound is added in each trial is 0.2 or 0.8 is a secret.

Mechanisms to Protect Attribute Privacy. Our definitions allow an analyst to specify secrets about global properties of a dataset that they wish to protect. In order to satisfy these definitions the analyst can use a general tool for providing Pufferfish privacy called the Wasserstein mechanism proposed by Song *et al.* [122]. However, this mechanism is computationally expensive and may require computing an exponential number of pairwise Wasserstein distances, which is not feasible in most practical settings. To this end, we propose efficient mechanisms in the following two settings.

For dataset attribute privacy, we consider a special class of functions and attribute properties and propose a mechanism based on Gaussian noise. Though the nature of the noise is added from the same family of distributions as the differentially private Gaussian Mechanism, in Section 5.4 we articulate that the similarity between the two is based solely on the nature of the noise. We show that the mechanism can be applied to datasets where (1) attributes follow a multivariate Gaussian distribution and (2) the function to be computed on the data and the attribute property to be protected are linear in the number of records in the dataset (e.g., mean). We also relax the Gaussian assumption in Section 5.4.1. Note

that with the help of variational auto-encoders (VAEs) [123], one can obtain a Gaussian representation of the data even if a dataset does not come from a Gaussian distribution. Moreover, such disentangled representations can be based on interpretable attributes [109] that are easier for specifying which attributes require protection, particularly when the original data are complex (e.g., pixels on an image vs. the gender of the person in it).[1]

For distributional attribute privacy, we consider a model where dependencies between the attributes form a Bayesian network. This model helps us capture the extent to which a sensitive attribute parameter affects parameters of attributes in the query, and we add noise proportional to this influence. Although our mechanism is inspired by the Markov Quilt mechanism [122], the difference in settings prompts several changes, including a different metric for measuring influence between the variables.

Finally, we note that although [18] identified that "there is little focus in the literature on rigorous and formal privacy guarantees for business data", they leave "the challenging problem of developing Pufferfish instantiations and algorithms for general aggregate secrets" as future work.

5.1.2 Related work

Machine learning models have been shown to memorize and leak data used to train them, raising questions about the release and use of these models in practice. For example, membership attacks [62] show that models can leak whether certain records (e.g., patient data) were part of the training dataset or not. Attribute (or feature) privacy attacks, on the other hand, consider leakage of attribute values at an individual level [64, 65], and property inference attacks show that global properties about datasets can be leaked [70, 69, 77].

Differential privacy (DP) [8, 113] guarantees individual-level privacy when publishing an output computed on a database, by bounding the influence of any record on the output

[1]Though naive use of VAEs may not provide end-to-end privacy guarantees, it serves as an example that it is possible to obtain a representation of non-Gaussian data with interpretable Gaussian features. We leave it as an interesting open question on how to provide end-to-end privacy-preserving feature disentanglement.

and adding noise. Importantly, DP does not aim to protect population-level information, and was designed to learn global properties of a dataset without sacrificing individual privacy. DP does provide *group privacy* guarantees for groups of k correlated records, but these quantitative guarantees are only meaningful when k is small relative to the size of the dataset. Syntactically, DP guarantees that if any individual record were to be changed—including all attributes of that record—the result of the analysis would be approximately the same. For attribute privacy, we seek similar guarantees if an entire attribute of the dataset were to be changed—including all individuals' values for that attribute.

The Pufferfish framework [18] that we instantiate and describe in detail in the following sections, can be seen as a generalization of differential privacy that explicitly states the information that needs to be kept secret and the adversary's background knowledge about the data. Blowfish privacy [124] also allows one to express secrets and publicly known information about the data, but expressed as constraints on the data rather than distributions over data. We adapt the Markov Quilt Mechanism from [122], who also employ the Pufferfish framework [18] for private analysis of correlated data, although they focus on individual-level privacy. Our focus instead on privacy of dataset properties and distributions leads to a substantially different instantiation of the Pufferfish framework where the secrets are defined over attribute values rather than individual records in the dataset.

Research on algorithmic fairness has proposed several definitions formalizing the idea that machine learning models should not exhibit discrimination based on protected attributes (e.g., gender or race). Demographic parity formalizes fairness by requiring that a classifier's predicted label is independent of an individual's protected attributes. Our notion of dataset attribute privacy is a general framework where one can specify what information about attributes need to be protected, with attribute independence being one such scenario. However, our attribute privacy definitions would not be useful for satisfying other fairness notions that explicitly incorporate protected attributes, such as affirmative action or fairness through awareness [121]. Moreover, techniques proposed to obtain fair representations of

the training data [125, 110] have been shown to still leak sensitive attributes [64] when applied in the privacy context.

5.2 Preliminaries

Pufferfish Privacy. The Pufferfish privacy framework [18] consists of three components: a set of secrets S, a set of discriminative pairs $\mathcal{Q} \subseteq S \times S$, and a class of data distributions Θ. S is a set of possible facts about the database that we might wish to hide. $\mathcal{Q}$ is the set of secret pairs (s_i, s_j), $s_i, s_j \in S$, that we wish to be indistinguishable. The class of data distributions Θ can be viewed as a set of conservative assumptions about the underlying distribution that generates the database.

Definition 4 ((ϵ, δ)-Pufferfish Privacy [18, 122][2])**.** *A mechanism $\mathcal{M}$ is (ϵ, δ)-Pufferfish private in a framework $(S, \mathcal{Q}, \Theta)$ if for all $\theta \in \Theta$ with $X \sim \theta$, for all secret pairs $(s_i, s_j) \in \mathcal{Q}$ such that $P(s_i|\theta) \neq 0$ and $P(s_j|\theta) \neq 0$, and for all $T \subseteq Range(\mathcal{M})$, we have*

$$P_{\mathcal{M},\theta}(\mathcal{M}(X) \in T|s_i, \theta) \leq \exp(\epsilon) P_{\mathcal{M},\theta}(\mathcal{M}(X) \in T|s_j, \theta) + \delta.$$

The Wasserstein Mechanism proposed in [122] and defined formally in Section 5.5 is the first general mechanism for satisfying instantiations of Pufferfish privacy framework. It defines sensitivity of a function F as the maximum Wasserstein distance between the distribution of $F(X)$ given two different realizations of secrets s_i and s_j for $(s_i, s_j) \in \mathcal{Q}$. The mechanism then instantiates the Laplace mechanism by outputting $F(X)$ plus Laplace noise that scales with this sensitivity. Although this mechanism works in general for any instantiation of the Pufferfish framework, computing Wasserstein distance for all pairs of secrets is computationally expensive, and will typically not be feasible in practice.

Song *et al.* [122] also gave the Markov Quilt Mechanism for some special structures of data dependence. It is more efficient than the Wasserstein Mechanism and also guarantees

[2]The original definition [18] and the one considered in [122] is $(\epsilon, 0)$-Pufferfish. We extend the definition to (ϵ, δ)-Pufferfish in the natural way.

$(\epsilon, 0)$-Pufferfish privacy. The Markov Quilt Mechanism of [122] assumes that the entries in the input database Y form a Bayesian network, as defined below. These entires could either be: (1) the multiple attributes of a single record when the database contained only one record, or (2) the attribute values across multiple records for a single-fixed attribute when the database contained multiple attributes. Hence, the original Markov Quilt Mechanism could not accommodate correlations across multiple attributes in multiple records, as we study in this chapter. Full details about the algorithm is given in Algorithm 11.

Definition 5 (Bayesian Networks)**.** *A Bayesian network is described by a set of variables* $Y = \{Y_1, \ldots, Y_n\}$ *and a directed acyclic graph* $G = (Y, E)$ *whose vertices are variables in* Y*. The probabilistic dependence on* Y *included by the network can be written as:* $\Pr(Y_1, \ldots, Y_n) = \Pi_{i=1}^{n} \Pr(Y_i | \mathsf{parent}(Y_i))$.

Definition 6 (Variable-Max-Influence [122])**.** *The maximum influence of a variable* Y_i *on a set of variables* Y_A *under* Θ *is:*

$$e^{v}_{\Theta}(Y_A|Y_i) = \sup_{\theta \in \Theta} \max_{a,b \in \mathcal{Y}} \max_{y_A \in \mathcal{Y}^{card(Y_A)}} \log \frac{P(Y_A = y_A | Y_i = a, \theta)}{P(Y_A = y_A | Y_i = b, \theta)}.$$

The above definition measures influence of a *variable value on values of other variables*. One can compare this to Definition 12 used in the Attribute-Private Markov Quilt Mechanism, which instead measures max-influence of a parameter of the probability distribution of a variable on the distribution of parameters of other variables, as is needed in the attribute privacy setting.

Recall the definition of a Markov Quilt (Definition 11), which is used in this mechanism.

Algorithm 11 Markov Quilt Mechanism $(Y, F, \{S, \mathcal{Q}, \Theta\}, \epsilon)$ [122]

Input: database Y, L-Lipschitz query F, Pufferfish framework $\{S, \mathcal{Q}, \Theta\}$, privacy parameter ϵ.

for each Y_i **do**

Let $G_i := \{(Y_Q, Y_N, Y_R) : Y_Q \text{ is a Markov Quilt of } Y_i\}$

for all Y_Q (with Y_N, Y_R) in G_i **do**

if $e^v_\Theta(Y_Q|Y_i) < \epsilon$ **then**

Set $b(Y_Q) = \frac{|Y_N|}{\epsilon - e^v_\Theta(Y_Q|Y_i)}$.

else

Set $b(Y_Q) = \infty$.

end if

end for

Set $b_i = \min_{Y_Q \in G_i} b(Y_Q)$.

end for

Set $b_{\max} = \max_i b_i$.

Sample $Z \sim \text{Lap}(L \cdot b_{\max})$.

Return $F(Y) + Z$

The Markov Quilt Mechanism [122] given in Algorithm 11 guarantees $(\epsilon, 0)$-Pufferfish Privacy.

Accuracy. We will measure accuracy of our mechanisms with the following definition. For real-valued outputs, this definition says that the mechanism must output an answer that is at most an additive α away from the true answer with probability $1-\beta$. For vector-valued outputs, this notion can be naturally extended using the appropriate norm.

Definition 7 ((α, β)-accuracy)**.** *A mechanism $\mathcal{M}$ with real-valued outputs is (α, β)-accurate*

for a function F if for all databases X,

$$P(|\mathcal{M}(X) - F(X)| > \alpha) \leq \beta.$$

5.3 Attribute Privacy Definitions

Data model and representation. The dataset X contains n records, where each record consists of m attributes. We view the dataset X as an $n \times m$ matrix. In this chapter, we are interested in privacy of the *columns*, which represent attributes that a data owner wishes to protect. Thus we refer to the matrix X as $X = [X_1, \ldots, X_m]$, where X_i is the column vector related to the ith attribute (column). In contrast, traditional differential privacy [8, 113] is concerned with privacy of the *rows* of the dataset matrix. We let X_i^j denote ith attribute value for jth record.

Each record is assumed to be sampled i.i.d. from an unknown distribution, where attributes within a single record can be correlated (e.g., consider height and weight). We use $C \subseteq [m]$ to denote a set of indices of the sensitive attributes that require privacy protection (e.g., race and gender may be sensitive attributes; hair color may be non-sensitive). The data owner wishes to compute a function F over her dataset and release the value (or estimate of the value) $F(X)$ while protecting some information about the sensitive attributes.

Privacy notions. We distinguish between three kinds of attribute privacy, corresponding to three different types of information the data owner may wish to protect.

Individual attribute privacy protects X_i^j for sensitive attribute i when $F(X)$ is released. Note that differential privacy provides individual attribute privacy simultaneously for all individuals and all attributes [8], but does not protect against individual-level inferences from population-level statistics [114]. For example, if a DP result shows a correlation between lung disease and smoking, one may infer that a known-smoker in the dataset has an elevated likelihood of lung disease.

Dataset attribute privacy is applicable when the owner wishes to reveal $F(X)$ while protecting the value of some function $g(X_i)$ for sensitive attribute $i \in C$ (e.g., whether there were more Caucasians or Asians present in the dataset).

Distribution attribute privacy protects privacy of a parameter ϕ_i that governs the distribution of ith sensitive attribute in the underlying population from which the data are sampled.

The last two notions are the ones put forward in this chapter and studied in detail. The difference between them may be subtle depending on g and ϕ. For example, consider one setting where the sensitive attribute is binary and g is the fraction of records where this attribute is 1, and another setting where the sensitive attribute is a Bernoulli random variable with parameter ϕ. In this case, g can be seen as an estimate of ϕ based on a sample. The difference becomes particularly relevant in settings where privacy is required for realizations of the dataset that are unlikely under the data distribution, or settings with small datasets where g is a poor estimate of ϕ.

Formal framework for attribute privacy. The standard notion of differential privacy is not directly applicable to our setting since we are interested in protecting population-level information. Instead, we formalize our attribute privacy definitions using the Pufferfish privacy framework of Definition 4 by specifying the three components $(S, \mathcal{Q}, \Theta)$. The distributional assumptions of this framework are additionally useful for formalizing correlation across attributes.

Definition 8 (Dataset Attribute Privacy)**.** *Let $(X_1^j, X_2^j, \ldots, X_m^j)$ be a record with m attributes that is sampled from an unknown distribution $\mathcal{D}$, and let $X = [X_1, \ldots, X_m]$ be a dataset of n records sampled i.i.d. from $\mathcal{D}$ where X_i denotes the (column) vector containing values of ith attribute of every record. Let $C \subseteq [m]$ be the set of indices of sensitive attributes, and for each $i \in C$, let $g_i(X_i)$ be a function with codomain $\mathcal{U}^i$.*

A mechanism $\mathcal{M}$ satisfies (ϵ, δ)-dataset attribute privacy *if it is (ϵ, δ)-Pufferfish private*

for the following framework $(S, \mathcal{Q}, \Theta)$*:*

Set of secrets: $S = \{s_a^i := \mathbb{1}[g_i(X_i) \in \mathcal{U}_a^i] : \mathcal{U}_a^i \subseteq \mathcal{U}^i, i \in C\}$.

Set of secret pairs: $\mathcal{Q} = \{(s_a^i, s_b^i) \in S \times S, i \in C\}$.

Distribution: Θ *is a set of possible distributions* θ *over the dataset* X*. For each possible distribution* $\mathcal{D}$ *over records, there exists a* $\theta_{\mathcal{D}} \in \Theta$ *that corresponds to the distribution over* n *i.i.d. samples from* $\mathcal{D}$*.*

This definition defines each secret s_a^i as the event that $g_i(X_i)$ takes a value in a particular set $\mathcal{U}_a^i$, and the set of secrets S is the collection of all such secrets for all sensitive attributes. This collection may include all possible subsets of $\mathcal{U}^i$, or it may include only application-relevant events. For example, if all $\mathcal{U}_a^i$ are singletons, this corresponds to protecting any realization of $g_i(X_i)$. Alternatively, the data owner may only wish to protect whether $g_i(X_i)$ is positive or negative, which requires only $\mathcal{U}_a^i = (-\infty, 0)$ and $\mathcal{U}_b^i = [0, \infty)$. The set of secret pairs $\mathcal{Q}$ that must be protected includes all pairs of the events on the same sensitive attribute. The Pufferfish framework considers distributions θ over the entire dataset X, whereas we require distributions $\mathcal{D}$ over records. We resolve this by defining Θ to be the collection of distributions over datasets induced by the allowable i.i.d. distributions over records.

Determining which functions g_i to consider is an interesting question. For example, in [126] the authors show that it is tractable to check whether the output of certain classes of functions evaluated on a dataset reveals information about the output of another query evaluated on the same dataset. Hence, given a function F whose output a data owner wishes to release, the owner may consider either those g_i's about which F reveals information, or those for which verifying perfect privacy w.r.t. F is infeasible.

Definition 9 (Distributional Attribute Privacy)**.** *Let* $(X_1^j, X_2^j, \ldots, X_m^j)$ *be a record with* m *attributes that is sampled from an unknown distribution described by a vector of random*

variables $(\phi_1, \ldots, \phi_m)$*, where* ϕ_i *parameterizes the marginal distribution of* X_i^j *conditioned on the values of all* ϕ_k *for* $k \neq i$*. The* $(\phi_1, \ldots, \phi_m)$ *are drawn from a known joint distribution* P*, and each* ϕ_i *has support* Φ^i*. Let* $X = [X_1, \ldots, X_m]$ *be a dataset of* n *records sampled i.i.d. from the distribution described by* $(\phi_1, \ldots, \phi_m)$ *where* X_i *denotes the (column) vector containing values of* i*th attribute of every record. Let* $C \subseteq [m]$ *be the set of indices of sensitive attributes.*

A mechanism $\mathcal{M}$ *satisfies* (ϵ, δ)-distributional attribute privacy *if it is* (ϵ, δ)*-Pufferfish private for the following framework* $(S, \mathcal{Q}, \Theta)$*:*

Set of secrets: $S = \{s_a^i := \mathbb{1}[\phi_i \in \Phi_a^i] : \Phi_a^i \subset \Phi^i, i \in C\}$.

Set of secret pairs: $\mathcal{Q} = \{(s_a^i, s_b^i) \in S \times S, i \in C\}$.

Distribution: Θ *is a set of possible distributions* θ *over the dataset* X*. For each possible* $\phi = (\phi_1, \ldots, \phi_m)$ *describing the conditional marginal distributions for all attributes, there exists a* $\theta_\phi \in \Theta$ *that corresponds to the distribution over* n *i.i.d. samples from the distribution over records described by* ϕ*.*

This definition naturally parallels Definition 8, with the attribute-specific random variable ϕ_i taking the place of the attribute-specific function $g_i(X_i)$. Although it might seem natural for ϕ_i to define the *marginal* distribution of the ith attribute, this would not capture the correlation across attributes that we wish to study. Instead, ϕ_i defines the *conditional marginal* distribution of the ith attribute given all other $\phi_{\neq i}$, which does capture such correlation. This also allows the distribution θ over datasets to be fully specified given these parameters and the size of the dataset.

More specifically, we model attribute distributions using standard notion of Bayesian hierarchical modeling. The $(\phi_1, \ldots, \phi_m)$ can be viewed as a set of hyperparameters of the distributions of the attributes, and P as hyper-priors of the hyperparameters. The distribution P is captured in Θ, and the distribution of attribute X_i is governed by a realization

of the random variable ϕ_i. The ϕ_i describes the conditional marginal distribution for attribute i: it is the hyperparameter of the probability of X_i given hyperparameters of all other attributes $P(X_i|\phi_1, \ldots, \phi_{i-1}, \phi_{i+1}, \ldots, \phi_m)$. We make the "naive" conditional independence assumption that all attributes X_i are mutually independent conditional on the set of parameters $(\phi_1, \ldots, \phi_m)$, hence, $(\phi_1, \ldots, \phi_m)$ fully capture the distribution of a record. The "naive" conditional independence is a common assumption in probabilistic models, and naive Bayes is a simple example that employs this assumption.

Since both of our attribute privacy definitions are instantiations of the Pufferfish privacy framework, one could easily apply the Wasserstein Mechanism [122] to satisfy $(\epsilon, 0)$-attribute privacy for either of our definitions. However, as described in Section 5.2, implementing this mechanism requires computing Wasserstein distance between the conditional distribution on $F(X)$ for all pairs of secrets in $\mathcal{Q}$. Computing exact Wasserstein distance is known to be computationally expensive, and our settings may require exponentially many computations in the worst case. In the remainder of the chapter, we provide efficient algorithms that satisfy each of these privacy definitions, focusing on dataset attribute privacy in Section 5.4 and distributional attribute privacy in Section 5.4.2, before returning to the (inefficient) Wasserstein Mechanism in Section 5.5.

5.4 The Gaussian Mechanism for Dataset Attribute Privacy

In this section we consider dataset attribute privacy as introduced in Definition 8. In this setting, an analyst wants to publish a function F evaluated on her dataset X, but is concerned about an adversary observing $F(X)$ and performing a Bayesian update to make inferences about a protected quantity $g_i(X_i)$. We propose a variant of the Gaussian Mechanism [113] that satisfies dataset attribute privacy when $F(X)$ conditioned on $g_i(X_i)$ follows a Gaussian distribution, with constant variance conditioned on $g_i(X_i) = a$ for all a. Although this setting is more restrictive, it is still of practical interest. For example, it can be applied when X follows a multivariate Gaussian distribution and g_i and F are linear with respect

to the entries of X, as we show in the instantiation of our mechanism in Section 5.4.2. We also note that using variational auto-encoders (VAEs) [123, 109], it is possible to encode data from other distributions using a Gaussian representation with interpretable features. This would then allow an analyst to specify which latent features are deemed sensitive for the data, even if the original features are less descriptive (e.g., pixels on an image vs. the gender of the person in it).

5.4.1 Attribute-Private Gaussian Mechanism

Algorithm 12 presents the Attribute-Private Gaussian Mechanism for answering a real-valued query $F(X)$ while protecting the values of $g_i(X_i)$ for $i \in C$. Much like the Gaussian Mechanism for differential privacy [113], the Attribute-Private Gaussian Mechanism first computes the true value $F(X)$, and then adds a Gaussian noise term with mean zero and standard deviation that scales with the sensitivity of the function. However, *sensitivity* of F in the attribute privacy setting is defined with respect to each secret attribute X_i as,

$$\Delta_i F = \max_{\theta \in \Theta} \max_{(s_a^i, s_b^i) \in \mathcal{Q}} |\mathbb{E}\left[F(X)|s_a^i, \theta\right] - \mathbb{E}\left[F(X)|s_b^i, \theta\right]|. \tag{5.1}$$

This differs from the sensitivity notion used in differential privacy in two key ways. First, we are concerned with measuring changes to the value of $F(X)$ caused by changing secrets s_a^i corresponding to realizations of $g_i(X_i)$, rather than by changing an individual's data. Second, we assume our data are drawn from an unknown underlying distribution θ, so $F(X)$ is a random variable. Our attribute privacy sensitivity bounds the maximum change in posterior expected value of $F(X)$ in the worst case over all distributions and pairs of secrets for each attribute. We note that if $F(X)$ is independent of the protected attribute X_i, then $\Delta_i F = 0$ and no additional noise is needed for privacy. The Attribute-Private Gaussian Mechanism of Algorithm 12 further benefits from the inherent randomness of the output $F(X)$. In particular, it reduces the variance σ^2 of the noise added by the conditional

variance of $F(X)$ given $g_i(X_i)$ and θ, as the sampling noise can mask some of the correlation. Hence, privacy also comes for free if the function of interest has low correlation with the protected attributes.

Algorithm 12 can be easily extended to handle vector-valued queries with $F(X) \in \mathbb{R}^k$ and sensitive functions g_i over multiple attributes by changing $\Delta_i F$ in Equation (5.1) to be the maximum ℓ_2 distance rather than absolute value. Additionally, the noise adjustment for each attribute should be based on the conditional covariance matrix of $F(X)$ rather than the conditional variance.

Algorithm 12 Attribute-Private Gaussian Mechanism, APGM($X, F, \{g_i\}, C, \{S, \mathcal{Q}, \Theta\}, \epsilon, \delta$) for dataset attribute privacy.

Input: dataset X, query F, functions g_i for protected attributes $i \in C$, framework $\{S, \mathcal{Q}, \Theta\}$, privacy parameters ϵ, δ
Set $\sigma^2 = 0$, $c = \sqrt{2\log(1.25/\delta)}$.
for each $i \in C$ **do**
    Set $\Delta_i F = \max_{\theta\in\Theta} \max_{(s_a^i, s_b^i)\in\mathcal{Q}} |\mathbb{E}\left[F(X)|s_i^a\right] - \mathbb{E}\left[F(X)|s_i^b\right]|$.
    if $(c\Delta_i F/\epsilon)^2 - \min_{\theta\in\Theta} \mathrm{Var}(F(X)|g_i(X_i), \theta) \geq \sigma^2$ **then**
        Set $\sigma^2 = (c\Delta_i F/\epsilon)^2 - \min_{\theta\in\Theta} \mathrm{Var}(F(X)|g_i(X_i), \theta)$.
if $\sigma^2 > 0$ **then**
    Sample $Z \sim \mathcal{N}(0, \sigma^2)$.
    Return $F(X) + Z$.
else Return $F(X)$.

Theorem 22. *The Attribute-Private Gaussian Mechanism APGM($X, F, \{g_i\}, C, \{S, \mathcal{Q}, \Theta\}, \epsilon, \delta$) in Algorithm 12 is (ϵ, δ)-dataset attribute private when $F(X)|g_i(X_i)$ is Gaussian distributed for any $\theta \in \Theta$ and $i \in C$.*

Privacy follows from the observation that the summation of $F(X)$ and the Gaussian noise Z is Gaussian distributed conditioned on any secrets, and the probabilities of the output conditioned on any pairs of secrets have the same variance with mean difference $\Delta_i F$. Since we bound the ratio of the two probabilities caused by shifting this variable, the analysis reduces to the proof of Gaussian mechanism in differential privacy.

Proof. Fix any pair of secrets $(s_a^i, s_b^i) \in \mathcal{Q}$ for a fixed secret attribute X_i under any

$\theta \in \Theta$. Recall that s_a^i denotes the event that $g_i(X_i) \in \mathcal{U}_a^i$. Let $Z \sim \mathcal{N}(0, \sigma^2)$ denote the Gaussian noise added in Algorithm 12. We have $[\mathcal{M}(X)|s_a^i, \theta] = [(F(X) + Z)|s_a^i, \theta] = [F(X)|s_a^i, \theta] + Z$, because Z is independent of s_a^i and θ. Since we have assumed that $F(X)|g_i(X_i)$ is Gaussian distributed and the summation of two Gaussians is Gaussian, $\mathcal{M}(X)|s_a^i$ follows a Gaussian distribution with mean $\mathbb{E}[F(X)|s_a^i, \theta]$ and variance $\mathrm{Var}(F(X)|g_i(X_i), \theta) + \sigma^2$. The ratio of probabilities of seeing an output w on a pair of secrets (s_a^i, s_b^i) in the worst case is as follows,

$$\begin{aligned}
&\max_{(s_a^i, s_b^i) \in \mathcal{Q}} |\log \frac{\exp(-\frac{1}{2}(\mathrm{Var}(F(X)|g_i(X_i), \theta) + \sigma^2)(w - \mathbb{E}[F(X)|s_i^a, \theta])^2}{\exp(-\frac{1}{2}(\mathrm{Var}(F(X)|g_i(X_i), \theta) + \sigma^2)(w - \mathbb{E}\left[F(X)|s_i^b, \theta\right])^2}| \\
=&|\log \frac{\exp(-\frac{1}{2}(\mathrm{Var}(F(X)|g_i(X_i), \theta) + \sigma^2)w^2)}{\exp(-\frac{1}{2}(\mathrm{Var}(F(X)|g_i(X_i), \theta) + \sigma^2)(w + \Delta)}|, \qquad (5.2)
\end{aligned}$$

where $\Delta = \max_{(s_i^a, s_i^b) \in \mathcal{Q}} \mathbb{E}[F(X)|s_a^i, \theta] - \mathbb{E}[F(X)|s_b^i, \theta]$. Equation (5.2) follows from shifting the variable w to $w + \mathbb{E}[F(X)|s_a^i, \theta]$. We observe that the probability ratio can be viewed as the probability ratio in the Gaussian Mechanism in *differential privacy* with noise draw from $\mathcal{N}(0, \mathrm{Var}(F(X)|g_i(X_i), \theta) + \sigma^2)$, and query sensitivity $\max_{(s_a^i, s_b^i) \in \mathcal{Q}} \mathbb{E}[F(X)|s_a^i] - \mathbb{E}[F(X)|s_b^i]$. Then, our analysis reduces to the proof of the Gaussian Mechanism in *differential privacy*. The Gaussian Mechanism for *differential privacy* with

$$\begin{aligned}
&\mathrm{Var}(F(X)|g_i(X_i), \theta) + \sigma^2 \\
\geq& 2\log(1.25/\delta) \left(\frac{\max_{(s_a^i, s_b^i) \in \mathcal{Q}} \mathbb{E}[F(X)|s_a^i, \theta] - \mathbb{E}[F(X)|s_b^i, \theta]}{\epsilon} \right)^2
\end{aligned}$$

ensures that with probability at least $1 - \delta$, we have

$$P(\mathcal{M}(X) \in T|s_a^i, \theta) \leq \exp(\epsilon) P(\mathcal{M}(X) \in T|s_b^i, \theta) + \delta,$$

for any $T \subseteq Range(\mathcal{M})$. Equivalently, we have

$$\sigma^2 \geq 2\log(1.25/\delta)\left(\frac{\max_{(s_a^i, s_b^i)\in\mathcal{Q}} \mathbb{E}\left[F(X)|s_a^i, \theta\right] - \mathbb{E}\left[F(X)|s_b^i, \theta\right]}{\epsilon}\right)^2 - \mathrm{Var}(F(X)|g_i(X_i), \theta).$$

Taking the maximum over i for all secret attributes and over all $\theta \in \Theta$, we require

$$\sigma^2 \geq \max_{i\in C}\left[2\log(1.25/\delta)(\Delta_i F/\epsilon)^2 - \min_{\theta\in\Theta}\mathrm{Var}(F(X)|g_i(X_i), \theta)\right]$$

which will ensure that the ratio of the probabilities that the algorithm $\mathcal{M}(X)$ outputs a query value in any subset T on any pair of secrets for any $\theta \in \Theta$ is bounded by ϵ with probability at least $1 - \delta$. □

High probability additive accuracy bounds on the output of Algorithm 12 can be derived using tail bounds on the noise term Z based on its variance σ^2. The formal accuracy guarantee is stated in Theorem 23, which follows immediately from tail bounds of a Gaussian.

Theorem 23. *The Attribute-Private Gaussian Mechanism*

APGM($X, F, \{g_i\}, C, \{S, \mathcal{Q}, \Theta\}, \epsilon, \delta$) in Algorithm 12 is (α, β)-accurate for any $\beta > 0$ and

$$\alpha = \sqrt{\max\{0, \max_{i\in C}\{(c\Delta_i F/\epsilon)^2 - \min_{\theta\in\Theta}\mathrm{Var}(F(X)|g_i(X_i), \theta)\}\}}\Phi^{-1}(1 - \frac{\beta}{2}),$$

where $c = \sqrt{2\log(1.25/\delta)}$ and Φ is the CDF of the standard normal distribution.

In general, if $F(X)$ is independent of, or only weakly correlated with the protected functions $g_i(X_i)$, then no noise is needed is preserve dataset attribute privacy, and the mechanism can output the exact answer $F(X)$. On the other hand, if $F(X)$ is highly correlated with $g_i(X_i)$, we then consider a tradeoff between the sensitivity and the variance of $F(X)$. If the variance of $F(X)$ is relatively large, then $F(X)$ is inherently private, and less noise

is required. If the variance of $F(X)$ is small and the sensitivity of $F(X)$ is large, the mechanism must add a noise term with large σ^2, resulting in low accuracy with respect to the true answer. To make these statements more concrete and understandable, Section 5.4.2 provides a concrete instantiation of Algorithm 12.

Privacy guarantees without Gaussian assumptions

A natural question is what privacy guarantee we can offer when our distributional assumption is violated. The following theorem states that if we can instead use Gaussian distributions that are close to the true distributions of $F(X)$ conditioned on $g_i(X_i)$ in Algorithm 12, then the loss in privacy is not too large. We quantify this distributional closeness using max-divergence.

Definition 10 (Max-Divergence)**.** *Let p and q be two distributions with the same support. The max-divergence $D(p||q)$ between them is defined as:*

$$D(p||q) = \sup_{T \subset \textit{Support}(p)} \log \frac{\Pr(p(x) \in T)}{\Pr(q(x) \in T)}.$$

The δ-approximate max-divergence between then is defined as:

$$D^{\delta}(p||q) = \sup_{T \subset \textit{Support}(p): \Pr[p(x) \in T] \geq \delta} \log \frac{\Pr(p(x) \in T) - \delta}{\Pr(q(x) \in T)}.$$

Theorem 24. *Let f_a^i denote the true distributon of $F(X)$ conditioned on the secret $s_a^i := \{g_i(X_i) = a\}$ for $i \in C$ and $s_a^i \in S$. For each f_a^i, if there exists a Gaussian distribution $\tilde{f}_a^i$ that is close to f_a^i with $\max\{D^{\delta_2}(f_a^i(x)||\tilde{f}_a^i(x)), D^{\delta_2}(\tilde{f}_a^i(x)||f_a^i(x))\}$ bounded above by D_{δ_2} for any $\delta_2 > 0$, then the Attribute-Private Gaussian Mechanism APGM($X, F, \{g_i\}, C, \{S, \mathcal{Q}, \Theta\}, \epsilon, \delta$) in Algorithm 12 with $\tilde{f}_a^i$s as the hypothesized distributions is $(\epsilon + 2D_{\delta_2}, \exp(D_{\delta_2})\delta + \delta_2)$-dataset attribute private.*

Proof. We first analyze the probability of the output conditioned on the event that

$\{\max_{i\in C}\max_{s_a^i\in S}\max\{D(f_a^i(x)||\tilde{f}_a^i(x)), D(\tilde{f}_a^i(x)||f_a^i(x))\} \leq D_{\delta_2}\}$. Fix any pair of secrets $(s_a^i, s_b^i) \in \mathcal{Q}$ for a fixed sensitive attribute X_i under any $\theta \in \Theta$. Let $Z \sim \mathcal{N}(0, \sigma^2)$ denote the Gaussian noise added in Algorithm 12. Let us partition $\mathbb{R}$ as $\mathbb{R} = R_1 \cup R_2$, where

$$R_1 = \{F(X) + Z \in \mathbb{R} : |F(X) + Z| \leq c\Delta_i F/\epsilon\},$$

and

$$R_2 = \{F(X) + Z \in \mathbb{R} : |F(X) + Z| > c\Delta_i F/\epsilon\}.$$

Fix any subset $T \subseteq \mathbb{R}$, and define $T_1 = T \cap R_1$ and $T_2 = T \cap R_2$.

For any $w \in T_1$, we can write the probability ratio of seeing the output w as follows:

$$\begin{aligned}
&\frac{\Pr(F(X) + Z = w|s_a^i, \theta)}{\Pr(F(X) + Z = w|s_b^i, \theta)} \\
=&\frac{\Pr(F(X) + Z = w|F(X)_{|s_a^i,\theta} \sim f_a^i)}{\Pr(F(X) + Z = w|F(X)_{|s_a^i,\theta} \sim \tilde{f}_a^i)} \cdot \frac{\Pr(F(X) + Z = w|F(X)_{|s_b^i,\theta} \sim \tilde{f}_b^i)}{\Pr(F(X) + Z = w|F(X)_{|s_b^i,\theta} \sim f_b^i)} \\
&\cdot \frac{\Pr(F(X) + Z = w|F(X)_{|s_a^i,\theta} \sim \tilde{f}_a^i)}{\Pr(F(X) + Z = w|F(X)_{|s_b^i,\theta} \sim \tilde{f}_b^i)}.
\end{aligned} \tag{5.3}$$

For any $w \in T_1$, the Attribute-Private Gaussian Mechanism ensures that the last ratio in Equation (5.3) is bounded above by $\exp(\epsilon)$. For the first ratio in Equation (5.3), since the generation process for z is indepedent of the data, we have

$$\begin{aligned}
&\frac{\Pr(F(X) + Z = w|F(X)_{|s_a^i,\theta} \sim f_a^i)}{\Pr(F(X) + Z = w|F(X)_{|s_a^i,\theta} \sim \tilde{f}_a^i)} \\
=&\frac{\int_f \Pr(Z = w - f)\Pr(F(X) = f|F(X)_{|s_a^i,\theta} \sim f_a^i)df}{\int_f \Pr(Z = w - f)\Pr(F(X) = f|F(X)_{|s_a^i,\theta} \sim \tilde{f}_a^i)df} \\
\leq&\max_f \frac{\Pr(F(X) = f|F(X)_{|s_a^i,\theta} \sim f_a^i)}{\Pr(F(X) = f|F(X)_{|s_a^i,\theta} \sim \tilde{f}_a^i)} \\
\leq&\exp(D_{\delta_2})
\end{aligned}$$

Similarly, we can bound the second ratio by $\exp(D_{\delta_2})$. Thus, Equation (5.3) is bounded by

$\exp(\epsilon + 2D_{\delta_2})$, which is equivalent to

$$\Pr(F(X) + Z \in T_1 | s_a^i, \theta) \leq \exp(\epsilon + 2D_{\delta_2}) \Pr(F(X) + Z \in T_1 | s_b^i, \theta) \tag{5.4}$$

We also bound the probability that the output belongs to the subset T_2 as follows:

$$\begin{aligned} \Pr(F(X) + Z \in T_2 | s_a^i, \theta) \leq & \exp(D_{\delta_2}) \Pr(F(X) + Z \in T_2 | F(X)_{|s_a^i, \theta} \sim \tilde{f}_a^i) \\ \leq & \exp(D_{\delta_2})\delta. \end{aligned} \tag{5.5}$$

Then, by (5.4) and (5.5), we have

$$\begin{aligned} & \Pr(F(X) + Z \in T | s_a^i, \theta) \\ = & \Pr(F(X) + Z \in T_1 | s_a^i, \theta) + \Pr(F(X) + Z \in T_2 | s_a^i, \theta) \\ \leq & \exp(\epsilon + 2D_{\delta_2}) \Pr(F(X) + Z \in T_1 | s_b^i, \theta) + \exp(D_{\delta_2})\delta. \end{aligned} \tag{5.6}$$

We then analyze the probability of the output when $\max_{i \in C} \max_{s_a^i \in S} \max\{D(f_a^i(x) || \tilde{f}_a^i(x)), D(\tilde{f}_a^i(x) || f_a^i(x))\}$ is bounded by D_{δ_2} with probability at least $1 - \delta_2$. For any $\delta_2 > \bar{\delta}$, define this hight probability event as follows:

$$E_{\delta_2} := \{\max_{i \in C} \max_{s_a^i \in S} \max\{D(f_a^i(x) || \tilde{f}_a^i(x)), D(\tilde{f}_a^i(x) || f_a^i(x))\} \leq D_{\delta_2}\}.$$

Let $E_{\delta_2}^c$ denote the complement set. By the choice of D_{δ_2}, we have $\Pr[E_{\delta_2}^c] \leq \delta_2$. Then by

(5.6) and the observation that $\Pr[E_{\delta_2}^c] \leq \delta_2$, we have that for any subset T,

$$
\begin{aligned}
&\Pr[F(X) + Z \in T | s_a^i, \theta] \\
\leq &\Pr[F(X) + Z \in T | s_a^i, \theta, E_{\delta_2}] \Pr[E_{\delta_2}] + \Pr[E_{\delta_2}^c] \\
\leq &(\exp(\epsilon + 2D_{\delta_2}) \Pr[F(X) + Z \in T | s_b^i, \theta, E_{\delta_2}] + \exp(D_{\delta_2})\delta) \Pr[E_{\delta_2}] \\
&+ \Pr[E_{\delta_2}^c] \\
= &\exp(\epsilon + 2D_{\delta_2}) \Pr[F(X) + Z \in T | s_b^i, \theta, E_{\delta_2}] \Pr[E_{\delta_2}] \\
&+ \exp(D_{\delta_2})\delta \Pr[E_{\delta_2}] + \Pr[E_{\delta_2}^c] \\
\leq &\exp(\epsilon + 2D_{\delta_2}) Pr[F(X) + Z \in T | s_b^i, \theta] + \exp(D_{\delta_2})\delta + \delta_2,
\end{aligned}
$$

completing the proof. □

5.4.2 Instantiation with Gaussian distributed data

In this section, we show an instantiation of our Attribute-Private Gaussian Mechanism when the joint distribution of the m attributes is multivariate Gaussian. The privacy guarantee of this mechanism requires that $F(X)|g_i(X_i)$ is Gaussian distributed, which is satisfied when g_i and F are linear with respect to the entries of X. For simplicity of illustration, we will choose both $F(X)$ and all $g_i(X_i)$ to compute averages.

As a motivating example, consider a dataset that consists of students' SAT scores X_{s}, heights X_{h}, weights X_{w}, and their family income X_{i}. As a part of a wellness initiative, the school wishes to release the average weight of its students, so $F(X) = \frac{1}{n}\sum_{j=1}^{n} X_{\mathsf{w}}^j$. The school also wants to prevent an adversary from inferring the average SAT scores or family income of their students, so $C = \{\mathsf{s}, \mathsf{i}\}$ and $g(X_i) = \frac{1}{n}\sum_{j=1}^{n} X_i^j$ for $i \in C$.

To instantiate our framework, let s_a^i denote the event that $g(X_i) = a$, which means the average value of column X_i is a. If $g(X_i)$ has support $\mathcal{U}^i$, then the set of secrets is $S = \{s_a^i : a \in \mathcal{U}^i, i \in C\}$, and the set of secret pairs is $\mathcal{Q} = \{(s_a^i, s_b^i) : a, b \in \mathcal{U}^i, a \neq b, i \in C\}$. Each $\theta \in \Theta$ is a distribution over n i.i.d. samples from an underlying multivariate

Gaussian distribution with mean $(\mu_1, \ldots, \mu_m)^T$ and covariance matrix (V_{ij}), $i, j \in [m]$, where $V_{ij} = V_{ji}$ is the covariance between X_i and X_j if $i \neq j$, and V_{ii} is the variance of X_i.

Suppose we want to guarantee (ϵ, δ)-dataset attribute privacy through the Attribute-Private Gaussian Mechanism. Then we need to first compute $\mathbb{E}\left[F(X)|s_a^i\right]$ and $\mathrm{Var}(F(X)|s_a^i)$ for each $i \in C$. Let j denote the index of the attribute averaged in $F(X)$. By the properties of a multivariate Gaussian distribution, the distribution of $F(X)$ conditional on $g(X_i) = a$ is Gaussian $\mathcal{N}(\bar{\mu}_a, \bar{V})$, where $\bar{\mu}_a = \mu_j + \frac{V_{ij}}{V_{ii}}(a - \mu_i)$ and $\bar{V} = \frac{1}{n}(V_{jj} - \frac{V_{ij}^2}{V_{ii}})$. We define the diameter of $\mathcal{U}$ as $d(\mathcal{U}) = \max_{a,b\in\mathcal{U}}|a - b|$. The sensitivity is: $\Delta_i F = \max_{(s_a^i, s_b^i)\in\mathcal{Q}}|\bar{\mu}_a - \bar{\mu}_b| = \frac{V_{ij}}{V_{ii}} \max_{a,b\in\mathcal{U}}|a - b| = \frac{V_{ij}}{V_{ii}} d(\mathcal{U})$. To ensure (ϵ, δ)-dataset attribute privacy for protected attribute X_i, the variance of the Gaussian noise must be at least $(c\frac{V_{ij}d(\mathcal{U})}{V_{ii}\epsilon})^2 - \frac{1}{n}(V_{jj} - \frac{V_{ij}^2}{V_{ii}})$ for $c = \sqrt{2\log(1.25/\delta)}$ as in Algorithm 12. Adding Gaussian noise with variance $\sigma^2 = \max_{i\in C}\{(c\frac{V_{ij}d(\mathcal{U})}{V_{ii}\epsilon})^2 - \frac{1}{n}(V_{jj} - \frac{V_{ij}^2}{V_{ii}})\}$ will provide (ϵ, δ)-dataset attribute privacy for all protected attributes.

We note that σ^2 is monotonically increasing with respect to V_{ij}. That is, our Attribute-Private Gaussian Mechanism will add less noise to the output if the query F is about an attribute which has a low correlation with the protected attributes.

So far we have discussed about the case when Θ only consists of one distribution, in order to show the impact of V_{ij}. For the general case, the sensitivity $\Delta_i F$ is $\max_{\theta\in\Theta} \frac{V_{ij}}{V_{ii}} d(\mathcal{U})$, and the noise is scaled with variance $\sigma^2 = \max_{i\in C}\{(c \max_{\theta\in\Theta} \frac{V_{ij}d(\mathcal{U})}{V_{ii}\epsilon})^2 - \min_{\theta\in\Theta} \frac{1}{n}(V_{jj} - \frac{V_{ij}^2}{V_{ii}})\}$.

sectionThe Markov Quilt Mechanism for Distributional Attribute Privacy

In this section we consider distributional attribute privacy, as introduced in Definition 9, and develop a mechanism that satisfies this privacy definition. Recall that in this setting, an analyst aims to release $F(X)$ while protecting the realization of a random parameter ϕ_i, which describes the conditional marginal distribution of the ith attribute, given the realization of all ϕ_k for $k \neq i$ for all other attributes. This formalization implies that all (column) attribute vectors X_i are mutually independent, conditional on the set of parame-

ters $(\phi_1, \ldots, \phi_m)$.

5.4.3 Attribute-Private Markov Quilt Mechanism

We base our mechanism on the idea of a *Markov Quilt*, which partitions a network of correlated random variables into those which are "near" (X_N) a particular variable X_i, and those which are "remote" (X_R). Intuitively, we will use this to partition attributes into those which are highly correlated (X_N) with our sensitive attributes, and those which are only weakly correlated (X_R).

Definition 11 (Markov Quilt)**.** *A set of nodes X_Q in a Bayesian network $G = (X, E)$ is a Markov Quilt for a node X_i if deleting X_Q partitions G into parts X_N and X_R such that $X_i \in X_N$ and X_R is independent of X_i conditioned on X_Q.*

We quantify the effect that changing the distribution parameter ϕ_i of a sensitive attribute X_i has on a set of distribution parameters ϕ_A (corresponding to a set of attributes X_A) using the *max-influence*. Since attributes are mutually independent conditioned on the vector $(\phi_1, \ldots, \phi_m)$, the max-influence is sufficient to quantify how much a change of all values in attribute X_i will affect the values of X_A. If ϕ_i and ϕ_A are independent, then X_A and X_i are also independent, and the max-influence is 0.

Definition 12. *The* max-influence *of an attribute X_i on a set of attributes X_A under Θ is:*

$$e_\Theta(X_A|X_i) = \sup_{\theta\in\Theta} \max_{\phi_i^a,\phi_i^b\in\Phi_i} \max_{\phi_A\in\Phi_A} \log \frac{P(\phi_A|\phi_i^a,\theta)}{P(\phi_A|\phi_i^b,\theta)}.$$

The sensitivity of F with respect to a set of attributes $A \subseteq [m]$, denoted $\Delta_A F$, is defined as the maximum change that the value of $F(X)$ caused by changing all columns X_A. Formally, we say that two datasets X, X' are *A-column-neighbors* if they are identical except for the columns corresponding to attributes in A, which may be arbitrarily different. Then

$$\Delta_A F = \max_{X,X' \; A\text{-column-neighbors}} |F(X) - F(X')|.$$

Although changing X_A may lead to changes in other columns, these changes are governed by the max influence, and will not affect attributes that are nearly independent of X_i.

Observe that the event that X_R and X_i are independent conditional on X_N is equivalent to the event when ϕ_R and ϕ_i are independent conditional on ϕ_N, which is why we can define the Markov Quilt based on X_i. However, since the distribution of X_is are governed by ϕ_is, the max-influence score must be computed using ϕ_is rather than X_is.

The mechanism. We extend the idea of the Markov Quilt Mechanism in [122] to the attribute privacy setting as follows. Let $A \subseteq [m]$ be a set of attributes over which F is computed. For example, F may compute the average of a particular attribute or a regression on several attributes. At a high level, we add noise to the output of F scaled based on the sensitivity of F with respect to $X_N s$. However, when computing the sensitivity of F we only need to consider sensitivity of F with respect to $A \cap N$, i.e., the queried set of attributes A that are in the "nearby" set of the protected attribute. If the query F is about attributes that are all in the "remote" set X_{R_i} and the max-influence on the corresponding Markov quilt is less than the privacy parameter ϵ, then $\Delta_{A \cap N} F$ is simply 0 and the mechanism will not add noise to the query answer. We note that the original Markov Quilt Mechanism was designed for protecting individual privacy, or one attribute record where the dataset only consists of one person's data. To extend the idea to protect dataset-level attribute information, non-trivial new data model and the corresponding max-influence definition is required.

Theorem 25. *The Attribute-Private Markov Quilt Mechanism*
APMQM($X, F, A, C, \{S, \mathcal{Q}, \Theta\}, \epsilon$) in Algorithm 13 is $(\epsilon, 0)$-distributional attribute private.

Proof. First, consider the case when $G = \emptyset$, which means there are no Markov quilt partitions such that the max-influence score is less than ϵ. In this case, for a fixed secret attribute X_i, the mechanism will simply add Laplace noise scaled with $\Delta_A F$. The privacy follows from Laplace Mechanism in differential privacy.

Algorithm 13 Attribute-Private Markov Quilt Mechanism, APMQM($X, F, A, C, \{S, \mathcal{Q}, \Theta\}, \epsilon$) for distributional attribute privacy.

Input: dataset X, query F, index set of queried attributes A, index set of sensitive attributes C, framework $\{S, \mathcal{Q}, \Theta\}$, privacy parameter ϵ.
for each $i \in C$ **do**
  Set $b_i = \Delta_A F/\epsilon$.
  Set $G_i := \{(X_Q, X_N, X_R) : e_\Theta(X_Q|X_i) \leq \epsilon\}$ to be all possible Markov quilts of X_i with max-influence less than ϵ.
  if $G_i \neq \emptyset$ **do**
    for each $(X_Q, X_N, X_R) \in G_i$ **do**
      if $\Delta_{A\cap N} F/(\epsilon - e_\Theta(X_Q|X_i)) \leq b_i$ **then**
        Set $b_i = \Delta_{A\cap N} F/(\epsilon - e_\Theta(X_Q|X_i))$.
Sample $Z \sim \text{Lap}(\max_{i\in C} b_i)$.
Return $F(X) + Z$

Let us consider the case when $G \neq \emptyset$. Below we bound the probability distribution for a single outcome w. For the set of outcomes T, the proof can be extended to bound the integral of the probability distribution over the set T. We fix any pair of secrets $(s_a^i, s_b^i) \in \mathcal{Q}$ for a fixed secret attribute X_i under any $\theta \in \Theta$. Here s_a^i denotes the event that $\phi_i = a$, and we consider the more general case later. Let Z be the Laplace noise as generated in Algorithm 13. If there exists a Markov quilt for X_i, $i \in C$, with max-influence score less than ϵ, then for any attribute $X_j \in A$:

$$
\begin{aligned}
&\max_{a,b} \frac{P(\mathcal{M}(X) = w|\phi_i = a, \theta)}{P(\mathcal{M}(X) = w|\phi_i = b, \theta)} \\
=&\max_{a,b} \frac{P(F(X) + Z = w|\phi_i = a, \theta)}{P(F(X) + Z = w|\phi_i = b, \theta)} \\
=&\max_{a,b} \max_{R\cup Q} \frac{P(F(X) + Z = w|\phi_i = a, \phi_{R\cup Q} = \bar{v}, \theta)}{P(F(X) + Z = w|\phi_i = b, \phi_{R\cup Q} = \bar{v}, \theta)} \\
&\cdot \frac{P(\phi_{R\cup Q} = \bar{v}|\phi_i = a, \theta)}{P(\phi_{R\cup Q} = \bar{v}|\phi_i = b, \theta)} \\
=&\max_{a,b} \max_{R\cup Q} \frac{P(F(X) + Z = w|X_{R\cup Q} = x_{R\cup Q}, \phi_i = a, \theta)}{P(F(X) + Z = w|X_{R\cup Q} = x_{R\cup Q}, \phi_i = b, \theta)} \\
&\cdot \frac{P(X_{R\cup Q} = x_{R\cup Q}|\phi_{R\cup Q} = \bar{v}, \theta)}{P(X_{R\cup Q} = x_{R\cup Q}|\phi_{R\cup Q} = \bar{v}, \theta)} \cdot \frac{P(\phi_{R\cup Q} = \bar{v}|\phi_i = a, \theta)}{P(\phi_{R\cup Q} = \bar{v}|\phi_i = b, \theta)}, \qquad (5.7)
\end{aligned}
$$

where the final equality Equation (5.7) follows from the independence between $X_{R\cup Q}$ and ϕ_i given $\phi_{R\cup Q} = \bar{v}$, and $x_{R\cup Q}$ denotes a realization of the $X_{R\cup Q}$ columns. For a fixed

$X_{R\cup Q}$, $F(X)$ can vary by at most $\Delta_{A\cap N}F$, and therefore, the first ratio is bounded by $\exp(\epsilon - \exp(e_\Theta(X_Q|X_i)))$. The second ratio in Equation (5.7) is 1, and the third ratio in Equation (5.7) is bounded by $\exp(e_\Theta(X_Q|X_i))$. Then Equation (5.7) is bounded above by $\exp(\epsilon - \exp(e_\Theta(X_Q|X_i)))\exp(e_\Theta(X_Q|X_i)) = \exp(\epsilon)$. For the general case when $s_a^i := \mathbb{1}[\phi_i \in \Phi_a^i] : \Phi_a^i \subset \Phi^i$, similarly, for any Φ_a^i and Φ_b^i, we have,

$$\begin{aligned}
&\frac{P(\mathcal{M}(X) = w|\phi_i \in \Phi_a^i, \theta)}{P(\mathcal{M}(X) = w|\phi_i \in \Phi_b^i, \theta)} \\
\leq &\max_{R\cup Q} \frac{P(F(X) + Z = w|X_{R\cup Q} = x_{R\cup Q}, \phi_i \in \Phi_a^i, \theta)}{P(F(X) + Z = w|X_{R\cup Q} = x_{R\cup Q}, \phi_i \in \Phi_b^i, \theta)} \\
&\cdot \frac{P(\phi_{R\cup Q} = \bar{v}|\phi_i \in \Phi_a^i, \theta)}{P(\phi_{R\cup Q} = \bar{v}|\phi_i \in \Phi_b^i, \theta)} \qquad (5.8) \\
\leq &\max_{R\cup Q} \max_{a\in\Phi_a^i, b\in\Phi_b^i} \frac{P(F(X) + Z = w|X_{R\cup Q} = x_{R\cup Q}, \phi_i = a, \theta)}{P(F(X) + Z = w|X_{R\cup Q} = x_{R\cup Q}, \phi_i = b, \theta)} \\
&\cdot \max_{a\in\Phi_a^i, b\in\Phi_b^i} \frac{P(\phi_{R\cup Q} = \bar{v}|\phi_i = a, \theta)}{P(\phi_{R\cup Q} = \bar{v}|\phi_i = b, \theta)}. \qquad (5.9)
\end{aligned}$$

The first ratio in Equation (5.9) is bounded by $\exp(\epsilon - \exp(e_\Theta(X_Q|X_i)))$ and the second ratio is bounded by $\exp(e_\Theta(X_Q|X_i))$, so Equation (5.9) is bounded above by $\exp(\epsilon)$, and the theorem follows.

□

As before, the accuracy follows immediately from the tail bound on the noise term based on the Laplace distribution parameter. The accuracy depends on $A\cap N$ and $e_\Theta(X_Q|X_i)$, which measures the correlations between the sensitive attributes and the queried attributes.

Theorem 26. *The Attribute-Private Markov Quilt Mechanism*

APMQM($X, F, A, C, \{S, \mathcal{Q}, \Theta\}, \epsilon$) in Algorithm 13 is (α, β)-accurate for any $\beta > 0$ and

$$\alpha = \max\{\max_{i\in C} \min_{(X_Q, X_N, X_R)\in G_i} \Delta_{A\cap N}F/(\epsilon - e_\Theta(X_Q|X_i)), \Delta_A F/\epsilon\} \log(\frac{1}{2\beta}),$$

where $G_i := \{(X_Q, X_N, X_R) : e_\Theta(X_Q|X_i) \leq \epsilon\}$ is the set of all possible Markov quilts of X_i with max-influence less than ϵ.

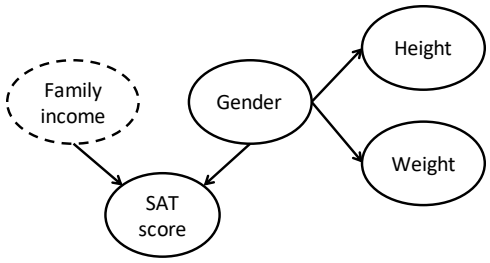


Figure 5.1: Bayesian Network of five attributes where income is a sensitive attribute.

Example 1. *Consider a dataset that consists of students' SAT scores* X_{s}*, heights* X_{h}*, weights* X_{w}*, gender* X_{g}*, and their family income* X_{i}*, where these variables form a Bayesian network as in Figure 5.1. The school wishes to release the number of students that are taller than* $5'6''$*, while protecting the distribution of family income of their students with privacy parameter* ϵ*. In this case,* $C = \{\mathsf{i}\}$*,* $A = \{\mathsf{h}\}$ *and* $F(X) = \sum_{j=1}^{n} \mathbb{1}[X_{\mathsf{h}}^j > 5'6'']$*. Consider a Markov quilt for* X_{i}*:* $Q = \{\mathsf{g}\}, N = \{\mathsf{i}, \mathsf{s}\}, R = \{\mathsf{h}, \mathsf{w}\}$*. Then* $A \cap N = \emptyset$*, so we can safely release* $F(X) = \sum_{i=1}^{n} \mathbb{1}[X_{\mathsf{h}}^j > 5'6'']$ *without additional noise.*

Next consider the case when the school wishes to release the number of students that are taller than $5'6''$ *and have SAT score* > 1300*. Then,* $F(X) = \sum_{j=1}^{n} \mathbb{1}[(X_{\mathsf{h}}^j > 5'6'') \wedge (X_{\mathsf{s}}^j > 1300)]$ *and* $A = \{\mathsf{h}, \mathsf{s}\}$*. In this case we can still use the same Markov quilt as before, but now* $A \cap N = \{\mathsf{s}\}$*. The mechanism will add Laplace noise scaled with* $\Delta_{\{\mathsf{s}\}} F/(\epsilon - e_{\Theta}(X_{\mathsf{g}}|X_{\mathsf{i}}))$*.*

It is instructive to contrast the above mechanism to the Markov Quilt Mechanism of [122], presented fully in Appendix **??**. The most important difference is that the mechanism in [122] was not designed to guarantee attribute privacy. It provides privacy of the values X_i^j but does not protect the distribution from which X_i^j is generated. This difference in high-level goals leads to three key technical differences. Firstly, the definition of max-influence in [122] measures influence of a *variable value on values of other variables*. This is insufficient when one wants to protect distributional information, as X_i may take a range of values while still following a particular distribution (e.g., hiding the gender of an individual in a dataset vs. hiding the proportion of females to males in this dataset.) Secondly, while it is natural to consider L-Lipschitz functions to bound sensitivity when one value

changes (as is done in [122]), this is not applicable to settings where the distribution of data changes, since this may change all values in a column. As a result, we do not restrict F in this way. Finally, the mechanisms themselves are different as [122] consider answering query F over all attributes of an individual. As a result, they need to consider sensitivity of a function to all the "nearby" attributes. In contrast, we only consider sensitivity of those "nearby" attributes that happen to be in the query (i.e., those in A).

5.5 The Wasserstein Mechanism for General Attribute Privacy

The Wasserstein Mechanism [122] (Algorithm 14) is a general mechanism for satisfying Pufferfish privacy; Algorithm 14 is $(\epsilon, 0)$-Pufferfish private for any instantiation of the Pufferfish framework [122]. It defines sensitivity of a function F as the maximum Wasserstein distance W_∞ between the distributions of $F(X)$ under two different realizations of secrets s_i and s_j for $(s_i, s_j) \in \mathcal{Q}$, and then outputs $F(X)$ plus Laplace noise that scales with this sensitivity. The distance metric W_∞ denotes the ∞-Wasserstein distance between two probability distributions, formally defined below.

Definition 13 (∞-Wasserstein distance, W_∞)**.** *Let μ, ν be two probability distributions on $\mathbb{R}$,*[3] *and let $\Gamma(\mu, \nu)$ be the set of all joint distributions with marginals μ and ν. The ∞-Wasserstein distance between μ and ν is defined as :*

$$W_\infty(\mu, \nu) = \inf_{\gamma \in \Gamma(\mu,\nu)} \max_{(x,y) \in support(\gamma)} |x - y|. \tag{5.10}$$

The ∞-Wasserstein distance is closely related to optimal transportation. Each $\gamma \in \Gamma(\mu, \nu)$ can be viewed as a way to shift probability mass between μ and ν, and the cost is $\max_{(x,y) \in support(\gamma)} |x - y|$. For discrete distributions, the ∞-Wasserstein distance is the minimum of the maximum distance that any probability mass moves to transform μ to ν.

[3] In general, Wasserstein distance can be defined on any metric space. We will use it only over the real numbers with the Euclidean metric.

Algorithm 14 Wasserstein Mechanism $(X, F, \{S, \mathcal{Q}, \Theta\}, \epsilon)$ [122]

Input: dataset X, query F, Pufferfish framework $\{S, \mathcal{Q}, \Theta\}$, privacy parameter ϵ

for all $(s_i, s_j) \in \mathcal{Q}$ and all $\theta \in \Theta$ such that $P(s_i|\theta) \neq 0$, and $P(s_j|\theta) \neq 0$ **do**

Set $\mu_{i,\theta} = P(F(X)|s_i, \theta)$, $\mu_{j,\theta} = P(F(X)|s_j, \theta)$.

Calculate $W_\infty(\mu_{i,\theta}, \mu_{j,\theta})$.

end for

Set $W = \sup_{(s_i,s_j)\in\mathcal{Q},\theta\in\Theta} W_\infty(\mu_{i,\theta}, \mu_{j,\theta})$.

Sample $Z \sim \text{Lap}(W/\epsilon)$.

Return $F(X) + Z$

Since our framework is an instantiation of Pufferfish privacy, the Wasserstein Mechanism provides a general way to protect either dataset attribute privacy or distributional attribute privacy, when instantiated with the appropriate Pufferfish framework $(S, \mathcal{Q}, \Theta)$. This is stated formally in Theorem 27 and illustrated in Examples 2 and 3 below.

Theorem 27. *The Wasserstein Mechanism $(X, F, \{S, \mathcal{Q}, \Theta\}, \epsilon)$ in Algorithm 14 is $(\epsilon, 0)$-dataset attribute private and $(\epsilon, 0)$-distributional attribute private.*

Despite the general purpose nature of the Wasserstein Mechanism for achieving attribute privacy, it is known that computing Wasserstein distance is computationally expensive [127]. Instantiating Algorithm 14 to satisfy attribute privacy may require computing Wasserstein distance for exponentially many pairs of secrets, one for each subset of values of $g_i(X_i)$ or ϕ_i. This motivates our study of the Attribute-Private Gaussian Mechanism (Algorithm 12) and the Attribute-Private Markov Quilt Mechanism (Algorithm 13), which are both computationally efficient for practical use.

Example 2 (Wasserstein Mechanism for Dataset Attribute Privacy)**.** *Consider two binary attributes X_1 and X_2, where X_1 is the non-sensitive attribute and X_2 is the sensitive attribute. Suppose the dataset contains data from four people, and let the underlying distribution and dependence between X_1 and X_2 is characterized by the following probability*

distributions:

$$P(X_1 = 1|X_2 = 1) = p_1 \text{ and } P(X_1 = 1|X_2 = 0) = p_2, \tag{5.11}$$

Suppose $0.4 \leq p_1 \leq 0.6$ *and* $0.4 \leq p_2 \leq 0.6$. *The analyst wishes to release the summation of* X_1, $F(X) = \sum_{j=1}^{4} X_1^j$, *while protecting the summation of* X_2, *so* $g(X_2) = \sum_{j=1}^{4} X_2^j$. *To instantiate our framework, let* s_a^2 *denote the event that* $g(X_2) = a$. *The support of* $g(X_2)$ *is* $\mathcal{U} = \{0, 1, 2, 3, 4\}$. *Then the set of secrets is* $S = \{s_a^2 : a \in \mathcal{U}\}$, *and the set of secret pairs is* $\mathcal{Q} = \{(s_a^2, s_b^2) : a, b \in \mathcal{U}, a \neq b\}$. *Each* $\theta \in \Theta$ *is a certain pair of* p_1 *and* p_2 *such that* $0.4 \leq p_1 \leq 0.6$ *and* $0.4 \leq p_2 \leq 0.6$.

Consider the pair of conditional probabilities $\mu_{a,\theta} = P(F(X) = \cdot|s_a^2, \theta)$ *and* $\mu_{b,\theta} = P(F(X) = \cdot|s_b^2, \theta)$. *The worst case Wasserstein distribution between the pair of conditional probability distributions is reached when* $p_1 = 0.4$, $p_2 = 0.6$, *and* $a = 0$, $b = 4$ *when the two conditional probabilities differ the most. We list the conditional probabilities for this case in Table 5.1.*

Table 5.1: Conditional probability distributions under two extreme secrets for dataset attribute privacy

j	0	1	2	3	4
$P(F(X) = j\|g(X_2) = 0)$	0.0256	0.1536	0.3456	0.3456	0.1296
$P(F(X) = j\|g(X_2) = 4)$	0.1296	0.3456	0.3456	0.1536	0.0256

Here, the Wasserstein distance $W_\infty(P(F(X)|g(X_2 = 0)), P(F(X)|g(X_2 = 4)) = 1$, *since the optimal transportation is moving the mass from* 1 *to* 2 *and* 4 *to* 3, *and the Wasserstein mechanism will add Lap*$(1/\epsilon)$ *noise to* $F(X)$.

The mechanism with group differential privacy would add Lap$(4/\epsilon)$, *which gives worse utility. We note that the noise we add depends largely on the underlying distribution class* Θ. *For example, when* $\Theta = \{0.3 \leq p_1, p_2 \leq 0.7\}$, *the worst case Wasserstein distance is 2 and the mechanism will add Lap*$(2/\epsilon)$ *noise to* $F(X)$. *When* $\Theta = \{0 \leq p_1, p_2 \leq 1\}$, *the worst case Wasserstein distance is 4, and the Wassertein mechanism will add the same*

amount of noise as group differential privacy.

Example 3 (Wasserstein Mechanism for Distributional Attribute Privacy)**.** *Consider the same setting as Example 2: a dataset of four people with two binary attributes X_1 and X_2, where X_1 is non-sensitive and X_2 is sensitive. Let the underlying distribution and dependence between realized attributes X_1 and X_2 still be governed by* (5.11)*, and for simplicity fix $p_1 = 0.4$ and $p_2 = 0.6$. In the setting of distributional attribute privacy, we are interested in the conditional marginal distribution parameters of X_i given the* parameter *for X_j, rather than the realization of X_j. We denote the Bernoulli distribution parameter for X_1 and X_2 as ϕ_1 and ϕ_2, respectively. According to* (5.11)*, we have $\phi_1 = 0.4\phi_2 + 0.6(1-\phi_2) = 0.6 - 0.2\phi_2$.*

The analyst wishes to release the summation of X_1, $F(X) = \sum_{j=1}^{4} X_1^j$, while protecting the distribution parameter ϕ_2 for X_2. To instantiate our framework, we let s_a^2 denote the event that $\phi_2 = a$, and we suppose the support of ϕ_2 is $\Phi^2 = [0.2, 0.8]$. The set of secrets is $S = \{s_a^2 : a \in \Phi^2\}$, and the set of secret pairs is $\mathcal{Q} = \{(s_a^2, s_b^2) : a, b \in \Phi^2, a \neq b\}$. Each $\theta \in \Theta$ is a certain pair of ϕ_1 and ϕ_2 such that $\Phi^2 = [0.2, 0.8]$ and $\phi_1 = 0.6 - 0.2\phi_2$.

In this case, the support for ϕ_1 is $[0.44, 0.56]$. Consider the pair of conditional probabilities $\mu_{a,\theta} = P(F(X) = \cdot | s_a^2, \theta)$ and $\mu_{b,\theta} = P(F(X) = \cdot | s_b^2, \theta)$. The worst case Wasserstein distribution between the pair of conditional probability distributions is reached when $a = 0.8$ and $b = 0.2$ when the two conditional probabilities differ the most. We list the conditional probabilities for this case in Table 5.2.

Table 5.2: Conditional probability distributions under two extreme secrets for distributional attribute privacy

j	0	1	2	3	4
$P(F(X) = j \mid \phi_2 = 0.8)$	0.0983	0.3091	0.3643	0.1908	0.0375
$P(F(X) = j \mid \phi_2 = 0.2)$	0.0375	0.1908	0.3643	0.3091	0.0983

The Wasserstein distance $W_\infty(P(F(X)|\phi_2 = 0.8), P(F(X)|\phi_2 = 0.2) = 1$, since the optimal transportation is moving the mass from 1 to 2 and 4 to 3, and the Wasserstein mechanism will add Lap$(1/\epsilon)$ noise to $F(X)$.

5.6 Conclusion

In this chapter, we study notions of privacy relevant for a data owner that releases statistics computed on her data. The notion encapsulates privacy of *dataset*- and *distribution*-level information and departs significantly from other notions studied in computer science that mostly relate to *individual* privacy. We also provide two efficient mechanisms and one inefficient but general mechanism that satisfy attribute privacy for these settings. We base our results on a novel use of the Pufferfish framework to account for correlations across attributes in the data, thus addressing" the challenging problem of developing Pufferfish instantiations and algorithms for general aggregate secrets" that was left open by[18]. Our mechanisms for satisfying the new privacy definitions would (1) encourage for more information about datasets to be released as they allow data owners to protect information they deem proprietary or sensitive, and (2) prevent inadvertent exposure of private data when information is released. Both of these can help improve the transparency of data-driven processes and make previously unavailable information more accessible.

Appendices

APPENDIX A
ADDITIONAL BASELINES AND NUMERICAL RESULTS FOR PAPRIKA(CHAPTER 3)

A.1 PrivLORD and PrivLORD2

In this section, we present two private versions of LORD++: PrivLORD in Algorithm 15 and PrivLORD2 in Algorithm 16. The former combines SPARSEVECTOR and LORD++, with the same threshold shifting as in PAPRIKA. The latter adds the candidacy checking step with constant λ on top of PrivLORD. The privacy of PrivLORD follows immediately from SPARSEVECTOR , and the privacy proof for PAPRIKA also applies to PrivLORD2 with a different choice of α_t.

Algorithm 15 PrivLORD($\alpha, W_0, \gamma, c, \epsilon, A$)

Input: stream of p-values $\{p_1, p_2, \ldots\}$ with mutiplicative sensitivity (η,μ), target FDR level α, initial wealth $W_0 < \alpha$, positive non-increasing sequence $\{\gamma_j\}_{j=0}^{\infty}$ of summing to one, expected number of rejections c, privacy parameters ϵ, threshold shift A.

Let $Z_\alpha^0 \sim \text{Lap}(2\eta c/\epsilon)$, count $= 0$

for each p-value p_t **do**

if count $\geq c$ **then** Output $R_t = 0$

else

Sample $Z_t \sim \text{Lap}(4\eta c/\epsilon)$.

if $t = 1$

then Set $\alpha_1 = \gamma_1 W_0$

else

Compute $\alpha_t = W_0\gamma_t + (\alpha - W_0)\gamma_{t-\tau_1} + \sum_{j\geq 2} \alpha\gamma_{t-\tau_j}$

if $\log p_t + Z_t \leq \log \alpha_t - A + Z_\alpha^{\text{count}}$

then Output $R_t = 1$. Set count = count $+1$ and sample $Z_\alpha^{\text{count}} \sim \text{Lap}(2\eta c/\epsilon)$

else Output $R_t = 0$

end for

Algorithm 16 PrivLORD2($\alpha, \lambda, W_0, \gamma, c, \epsilon, \delta, A$)

Input: stream of p-values $\{p_1, p_2, \ldots\}$ with mutiplicative sensitivity (η,μ), target FDR level α, candidacy threshold λ, initial wealth $W_0 < \alpha$, positive non-increasing sequence $\{\gamma_j\}_{j=0}^{\infty}$ of summing to one, expected number of rejections c, privacy parameters ϵ, δ, threshold shift A.

Let $Z_\alpha^0 \sim \text{Lap}(2\eta c/\epsilon)$, count $= 0$

for each p-value p_t **do**

if count $\geq c$ **then** Output $R_t = 0$

else

Sample $Z_t \sim \text{Lap}(4\eta c/\epsilon)$. Set the indicator for candidacy $C_t = I(\log p_t < \log \lambda)$.

if $t = 1$

then Set $\alpha_1 = \gamma_1 W_0$

else

Compute $\alpha_t = W_0\gamma_t + (\alpha - W_0)\gamma_{t-\tau_1} + \sum_{j\geq 2} \alpha\gamma_{t-\tau_j}$

if $C_t = 1$ and $\log p_t + Z_t \leq \log \alpha_t - A + Z_\alpha^{\text{count}}$

then Output $R_t = 1$. Set count = count $+1$ and sample $Z_\alpha^{\text{count}} \sim \text{Lap}(2\eta c/\epsilon)$

else Output $R_t = 0$

end for

A.2 Additional Numerical Results

Tables A.1 and A.2 report the numerical values for our experiments on Bernoulli and truncated exponential data, respectively. This information is also presented visually in Figures 3.1 and 3.2.

Table A.1: Numerical values of FDR and power for Bernoulli observations experiments. LapSAFFRON corresponds to running SAFFRON on the naïve Laplace privatization of the p-values.

π	ϵ	PAPRIKA AI		PAPRIKA		SAFFRON AI		SAFFRON		LORD		Alpha-investing		LapSAFFRON	
		FDR	power	FDR	power	FDR	power	FDR	power	FDR	power	FDR	power	FDR	power
0.01	3	0	.825	0	.817	0	.833	0	.833	0	.833	0	.833	.990	.485
	5	0	.833	0	.833										
	10	0	.833	0	.833										
0.02	3	0	.844	.017	.810	0	.938	0	.938	0	.938	0	.875	.973	.509
	5	0	.916	.001	.900										
	10	0	.941	0	.938										
0.03	3	.008	.457	.103	.389	.077	.923	0	.846	0	.846	0	.692	.977	.509
	5	.006	.694	.018	.670										
	10	.015	.849	.007	.808										
0.04	3	.003	.604	.120	.580	.030	.970	0	.879	0	.940	0	.848	.943	.512
	5	.003	.756	.035	.740										
	10	.060	.860	.008	.836										
0.05	3	.009	.560	.168	.514	.056	.971	.056	.971	.105	.971	.056	.971	.940	.505
	5	.007	.815	.053	.785										
	10	.017	.938	.012	.922										

Table A.2: Numerical values of FDR and power for truncated exponential observations experiments. LapSAFFRON corresponds to running SAFFRON on the naïve Laplace privatization of the p-values.

π	ϵ	PAPRIKA AI		PAPRIKA		SAFFRON AI		SAFFRON		LORD		Alpha-investing		LapSAFFRON	
		FDR	power	FDR	power	FDR	power	FDR	power	FDR	power	FDR	power	FDR	power
0.01	3	0	.995	0	.987	0	1.00	0	1.00	0	1.00	0	.638	.989	.543
	5	0	1.00	0	1.00										
	10	0	1.00	0	1.00										
0.02	3	0	.936	0	.903	0	1.00	0	1.00	0	.999	0	.676	.973	.505
	5	0	.994	0	.993										
	10	0	.999	0	1.00										
0.03	3	0	.708	.005	.618	0	1.00	0	1.00	0	1.00	0	.982	.977	.516
	5	0	.958	0	.942										
	10	0	.999	0	.996										
0.04	3	0	.569	.003	.474	0	1.00	0	1.00	0	1.00	0	.999	.944	.503
	5	0	.905	0	.873										
	10	0	.998	0	.996										
0.05	3	0	.394	.007	.327	0	1.00	0	1.00	0	1.00	0	1.00	.940	.505
	5	0	.825	.002	.726										
	10	0	.990	0	.986										

APPENDIX B

ADDITIONAL EXPERIMENTAL SETUP AND RESULTS FOR DATASET-LEVEL ATTRIBUTE LEAKAGE (CHAPTER 4)

B.1 Attribute Correlation in Datasets

In this section we provide information on how correlation cases were determined for the datasets and attributes in Table 4.3.

Health Dataset. We measure the correlations between `Gender` or `ClaimsTruncated` and the 133 categorical attributes and 6 numerical attributes by Cramer's V scores and point biserial correlation coefficients, respectively. With `Gender` as the sensitive attribute, we identify 22 categorical attributes that have Cramer's V scores greater than 0.15 and 2 numerical attributes that have point biserial correlation (absolute value) greater than 0.1. The attributes that have the highest Cramer's V are `sp10` (0.218), `noSpecialities` (0.212), `noProviders` (0.208), `noVendors` (0.201). To give a overview of correlations including weak correlation with other attributes, we identify 17 attributes that have Cramer's V scores within the range $[0.1, 0.15]$ and 37 attributes Cramer's V scores within the range $[0.5, 0.1]$. The Cramer's V score between DaysInHospital and `Gender` is 0.09, and thus, we deem them as uncorrelated.

With `ClaimsTruncated` as the sensitive attribute, we identify 50 categorical attributes (e.g., `sp1` (0.42), `sp2` (0.51), `pcg1` (0.41), etc.) that have Cramer's V scores greater than 0.15, and 4 numerical attributes that have point biserial correlation (absolute value) greater than 0.1. The Cramer's V score between `DaysInHospital` and `ClaimsTruncated` is 0.13, and we deem them as uncorrelated.

Table B.1: Correlation factors for the Adult dataset.

Attributes	Gender	Income
Cramer's V scores		
EducationLevel	0.042	0.326
MaritalStatus	0.466	0.448
Occupation	0.435	0.329
Relationship	0.650	0.454
Race	0.119	0.099
NativeCountry	0.059	0.096
Income	0.217	-
Gender	-	0.217
point biserial correlation coefficients		
Age	0.082	0.229
CapitalGain	0.049	0.221
CapitalLoss	0.047	0.150
HoursPerWeek	0.231	0.230

Adult Dataset. We measure the correlations between Gender or Income and the 7 categorical attributes and 4 numerical attributes by Cramer's V scores and point biserial correlation coefficients, respectively. We list all the correlation factors in Table B.1, as X only has 11 attributes. For Gender, we identify 4 categorical attributes that have Cramer's V scores above 0.15 and 1 numerical attribute that has point biserial correlation coefficients above 0.1. The sensitive attribute Income has a high Cramer's V score with 5 categorical attributes and the target variable EducationLevel, as well as high point biserial correlation coefficients with 4 numerical attributes.

Crime Dataset. Since all features are numerical, we measure the correlations by Pearson correlation coefficients. Table B.2 shows the number of attributes that have the coefficients within a certain range. We use 0.4 as the threshold to determine X'. The target variable CrimesPerPop is correlated with both TotalPctDivorce and Income, with corre-

Table B.2: The distribution of correlation factors for the Crime dataset.

Range	`TotalPctDivorce`	`Income`
$[0.5, 1]$	15	34
$[0.4, 0.5)$	11	4
$[0.3, 0.4)$	31	22
$[0.2, 0.3)$	4	12
$[0.1, 0.2)$	14	19

lation coefficients 0.553 and -0.424, respectively.

Yelp-Health and Amazon Datasets. For Yelp-Health dataset, the point biserial correlation coefficients between `Specialty` and `ReviewRating` is 0.009, hence, the scenario corresponds $X \sim A, Y \perp A$. The review text is clearly correlated with the doctor specialty as shown in Table 4 in [70].

For the Amazon dataset, since the `ProductType` has 4 levels, we use the ANOVA to test whether the differences between the means of `ReviewScore` across different product types are statistically significant. The ANOVA p-value is $7.6e-83$. We conjecture that the co-purchasing graph X is also correlated with the `ProductType`, and thus, the scenario corresponds $X \sim A, Y \sim A$.

B.2 Additional Results

In this section, we present additional attack results for Health, Adult and Crime dataset with smaller size of D_{aux} from Table 4.2. We show accuracies for both pooled model and the honest party's local model and the utility increase in Table B.4. Figure B.1 complements results in Section 4.6.4 on Amazon dataset trained with the sensitive attribute A.

B.3 White-box Attack Results

We also performed experiments where the attacker has access to the model parameters, i.e., the white box setting. As for the meta-classifier model, we use a two-layer network with

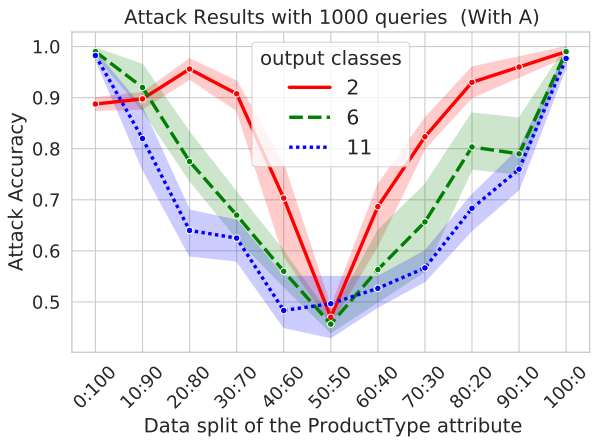


Figure B.1: Attack accuracy for the Amazon graph data when the sensitive attribute `ProductType` is used during training for different numbers of output classes across different distributions (splits).

Table B.3: Multi-Party Setting: Black-box attack accuracy for predicting the value of the distribution of sensitive variable A in the dataset of P_{honest}. We use smaller size of D_{aux} listed in Table 4.2, while all other settings are the same as in Table 4.4.

Datasets (Output Classes)	Model Type	Attack Accuracy		A	# X'
		A	$\bar{A}$		
Health (2)	Multi-layer Perceptron	.59	.55	`Gender`	24/139
		. 67	. 56	`ClaimsTruncated`	54/139
Adult (4)	Logistic Regression	.73	.76	`Gender`	5/11
		.84	.91	`Income`	9/11
Crime (3)	Multi-layer Perceptron	.60	.56	`TotalPctDivorce`	26/98
		.62	.60	`Income`	38/98

200 and 50 hidden units and learning rate 0.001. Each meta-classifier is trained based on 100 shadow models using Adam optimizer. Here, the meta-classifier takes as input model parameters as opposed to model inferences. Table B.5 shows the results. For logistic regression, the results are similar to those in Table 4.5 for the black-box setting. However, the attack accuracy for neural networks (MLP) reduces significantly. This was also noted in the work by [69]. One reason for this is that it is hard for a naive meta-classifier to learn the structure of equivalent symmetrical weights of neural networks. Indeed, one of the contributions of [69] is a technique for identifying this symmetry.

Table B.4: Test accuracies of the model trained on pooled dataset and the model trained only on honest party's data. The split in the honest party is $33:67$ based on the sensitive attribute.

Datasets (Output Classes)	Sensitive Attribute	Pooled Accuracy	Local Accuracy	Utility Increase
Health (2)	`Gender`	85.22%	84.64%	.58%
	`ClaimsTruncated`	76.63%	73.56%	3.07%
Adult (4)	`Gender`	73.23%	72.46%	.76%
	`Income`	71.14%	70.43%	.71%
Crime (3)	`TotalPctDivorce`	74.52%	72.35%	2.17%
	`Income`	72.81%	71.30%	1.51%
Yelp-Health (2)	`Specialty`	86.28%	80.38%	5.90%
Amazon (2)	`ProductType`	76.80%	76.28%	.62%
Amazon (6)	`ProductType`	45.92%	42.50%	3.42%
Amazon (11)	`ProductType`	27.94%	26.09%	1.85%

Table B.5: White-box attack accuracy for predicting whether the values of sensitive variable A in D_{honest}, the data of the honest party, are predominantly <5 or >5. The attack accuracy is evaluated on 100 D_{honest} datasets: half with 33:67 and half with 67:33 split and D_{adv} has 33:67 split. A synthetic correlation with A is added to the variables X and Y depending on the specific case. `R` corresponds to the setting where only 3 attributes are used for training instead of all data. Attack accuracy based on a random guess is 0.5.

Model	Logistic Regression				Neural Network			
Datasets	Adult		Health		Adult		Health	
Synthetic Variable	A	$\bar{A}$	A	$\bar{A}$	A	$\bar{A}$	A	$\bar{A}$
$X \sim A, Y \sim A$	.90	.94	.85	.97	.54	.49	.65	.61
$X \perp A, Y \sim A$	.95	.93	.81	.80	.57	.53	.63	.56
$X \sim A, Y \perp A$	.54	.53	.50	.53	.56	.53	.54	.51
$X \sim A, Y \perp A$ (`R`)	.75	.63	.76	.68	.55	.50	.55	.45

REFERENCES

[1] Wikipedia, "Facebook–cambridge analytica data scandal," *https://en.wikipedia.org/wiki/Facebook–Cambridge*$_A nalytica_d ata_s candal$,

[2] B. Auxier, L. Rainie, M. Anderson, A. Perrin, M. Kumar, and E. Turner, "Americans and privacy: Concerned, confused and feeling lack of control over their personal information," *https://www.pewresearch.org/internet/2019/11/15/americans-and-privacy-concerned-confused-and-feeling-lack-of-control-over-their-personal-information/*, 2019.

[3] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1322–1333.

[4] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2017, pp. 3–18.

[5] N. Carlini, C. Liu, J. Kos, Ú. Erlingsson, and D. Song, "The secret sharer: Measuring unintended neural network memorization & extracting secrets," 2018.

[6] S. Zanella-Béguelin, L. Wutschitz, S. Tople, V. Rühle, A. Paverd, O. Ohrimenko, B. Köpf, and M. Brockschmidt, "Analyzing information leakage of updates to natural language models," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 363–375.

[7] A. Narayanan and V. Shmatikov, "How to break anonymity of the netflix prize dataset," *arXiv preprint cs/0610105*, 2006.

[8] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proceedings of the 3rd Conference on Theory of Cryptography*, ser. TCC '06, 2006, pp. 265–284.

[9] Ú. Erlingsson, V. Pihur, and A. Korolova, "RAPPOR: Randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM Conference on Computer and Communications Security*, ser. CCS '14, New York, NY, USA: ACM, 2014, pp. 1054–1067.

[10] Differential Privacy Team, Apple, *Learning with privacy at scale*, https://machinelearning.apple.com/docs/learning-with-privacy-at-scale/appledifferentialprivacysystem.pdf, 2017.

[11] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *Advances in Neural Information Processing Systems 30*, ser. NIPS '17, Curran Associates, Inc., 2017, pp. 3571–3580.

[12] A. N. Dajani, A. D. Lauger, P. E. Singer, D. Kifer, J. P. Reiter, A. Machanavajjhala, S. L. Garfinkel, S. A. Dahl, M. Graham, V. Karwa, H. Kim, P. Lelerc, I. M. Schmutte, W. N. Sexton, L. Vilhuber, and J. M. Abowd, *The modernization of statistical disclosure limitation at the U.S. census bureau*, Presented at the September 2017 meeting of the Census Scientific Advisory Committee, 2017.

[13] R. Cummings, S. Krehbiel, Y. Mei, R. Tuo, and W. Zhang, "Differentially private change-point detection," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 825–10 834.

[14] W. Zhang, S. Krehbiel, R. Tuo, Y. Mei, and R. Cummings, "Single and multiple change-point detection with differential privacy," *Journal of Machine Learning Research*, vol. 22, no. 29, pp. 1–36, 2021.

[15] W. Zhang, G. Kamath, and R. Cummings, "Paprika: Private online false discovery rate control," *arXiv preprint arXiv:2002.12321*, 2020.

[16] W. Zhang, S. Tople, and O. Ohrimenko, "Dataset-level attribute leakage in collaborative learning," *arXiv preprint arXiv:2006.07267*, 2020.

[17] W. Zhang, O. Ohrimenko, and R. Cummings, "Attribute privacy: Framework and mechanisms," *arXiv preprint arXiv:2009.04013*, 2020.

[18] D. Kifer and A. Machanavajjhala, "Pufferfish: A framework for mathematical privacy definitions," *ACM Transactions on Database Systems (TODS)*, vol. 39, no. 1, pp. 1–36, 2014.

[19] C. Dwork, A. Roth, *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[20] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta, "Discovering frequent patterns in sensitive data," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 503–512.

[21] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum, "Differential privacy under continual observation," in *Proceedings of the 42nd ACM Symposium on Theory of Computing*, ser. STOC '10, 2010, pp. 715–724.

[22] G. Shmueli and H. Burkom, "Statistical challenges facing early outbreak detection in biosurveillance," *Technometrics*, vol. 52, no. 1, pp. 39–51, 2010.

[23] M. F. D'Angelo, R. M. Palhares, R. H. Takahashi, and R. H. Loschi, "Fuzzy/bayesian change point detection approach to incipient fault detection," *IET control theory & applications*, vol. 5, no. 4, pp. 539–551, 2011.

[24] J. Chen and A. K. Gupta, *Parametric statistical change point analysis: with applications to genetics, medicine, and finance*. Springer Science & Business Media, 2011.

[25] L. Lai, Y. Fan, and H. V. Poor, "Quickest detection in cognitive radio: A sequential change detection framework," in *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*, IEEE, 2008, pp. 1–5.

[26] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blažek, and H. Kim, "Detection of intrusions in information systems by sequential change-point methods," *Statistical methodology*, vol. 3, no. 3, pp. 252–293, 2006.

[27] A. G. Tartakovsky, A. S. Polunchenko, and G. Sokolov, "Efficient computer network anomaly detection by changepoint detection methods," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 1, pp. 4–11, 2012.

[28] R. Lund and J. Reeves, "Detection of undocumented changepoints: A revision of the two-phase regression model," *Journal of Climate*, vol. 15, no. 17, pp. 2547–2554, 2002.

[29] J. Bai and P. Perron, "Computation and analysis of multiple structural change models," *Journal of Applied Econometrics*, vol. 18, no. 1, pp. 1–22, 2003.

[30] N. Zhang and D. O. Siegmund, "Model selection for high-dimensional, multi-sequence change-point problems," *Statistica Sinica*, vol. 22, no. 4, pp. 1507–1538, 2012.

[31] W. A. Shewhart, *Economic Control of Quality of Manufactured Product*. D. Van Norstrand Company, Inc., 1931.

[32] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[33] A. N. Shiryaev, "On optimum methods in quickest detection problems," *Theory of Probability & Its Applications*, vol. 8, no. 1, pp. 22–46, 1963.

[34] S. W. Roberts, "A comparison of some control chart procedures," *Technometrics*, vol. 8, no. 3, pp. 411–430, 1966.

[35] G. Lorden, "Procedures for reacting to a change in distribution," *The Annals of Mathematical Statistics*, vol. 42, no. 6, pp. 1897–1908, 1971.

[36] M. Pollak, "Optimal detection of a change in distribution," *The Annals of Statistics*, vol. 13, no. 1, pp. 206–227, 1985.

[37] ——, "Average run lengths of an optimal method of detecting a change in distribution," *The Annals of Statistics*, vol. 15, no. 2, pp. 749–779, 1987.

[38] G. V. Moustakides, "Optimal stopping times for detecting changes in distributions," *The Annals of Statistics*, vol. 14, no. 4, pp. 1379–1387, 1986.

[39] T. L. Lai, "Sequential changepoint detection in quality control and dynamical systems," *Journal of the Royal Statistical Society, Series B*, vol. 57, no. 4, pp. 613–658, 1995.

[40] ——, "Sequential analysis: some classical problems and new challenges," *Statistica Sinica*, vol. 11, no. 2, pp. 303–408, 2001.

[41] M. Kulldorff, "Prospective time periodic geographical disease surveillance using a scan statistic," *Journal of the Royal Statistical Society, Series A*, vol. 164, no. 1, pp. 61–72, 2001.

[42] Y. Mei, "Sequential change-point detection when unknown parameters are present in the pre-change distribution," *The Annals of Statistics*, vol. 34, no. 1, pp. 92–122, 2006.

[43] ——, "Is average run length to false alarm always an informative criterion?" *Sequential Analysis*, vol. 27, no. 4, pp. 354–419, 2008.

[44] ——, "Efficient scalable schemes for monitoring a large number of data streams," *Biometrika*, vol. 97, no. 2, pp. 419–433, 2010.

[45] H. P. Chan, "Optimal sequential detection in multi-stream data," *The Annals of Statistics*, vol. 45, no. 6, pp. 2736–2763, 2017.

[46] D. V. Hinkley, "Inference about the change-point in a sequence of random variables," *Biometrika*, vol. 57, no. 1, pp. 1–17, 1970.

[47] E. Carlstein, "Nonparametric change-point estimation," *The Annals of Statistics*, vol. 16, no. 1, pp. 188–197, 1988.

[48] C. L. Canonne, G. Kamath, A. McMillan, A. Smith, and J. Ullman, "The structure of optimal private tests for simple hypotheses," in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, ser. STOC '19, 2019, pp. 310–321.

[49] A. W. Van Der Vaart and J. A. Wellner, *Weak convergence*. Springer, 1996, pp. 16–28.

[50] F. McSherry, "Privacy integrated queries: An extensible platform for privacy-preserving data analysis," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '09, 2009, pp. 19–30.

[51] S. National Academies, "Reproducibility and replicability in science," 2019.

[52] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig, "Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays," *PLoS genetics*, vol. 4, no. 8, e1000167, 2008.

[53] C. Dwork, W. J. Su, and L. Zhang, "Differentially private false discovery rate control," *arXiv preprint arXiv:1807.04209*, 2018.

[54] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: A practical and powerful approach to multiple testing," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 57, no. 1, pp. 289–300, 1995.

[55] J. Tian and A. Ramdas, "ADDIS: An adaptive discarding algorithm for online FDR control with conservative nulls," in *Advances in Neural Information Processing Systems 32*, ser. NeurIPS '19, Curran Associates, Inc., 2019, pp. 9383–9391.

[56] D. P. Foster and R. A. Stine, "α-investing: A procedure for sequential control of expected false discoveries," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 70, no. 2, pp. 429–444, 2008.

[57] E. Aharoni and S. Rosset, "Generalized α-investing: Definitions, optimality results and application to public databases," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 76, no. 4, pp. 771–794, 2014.

[58] A. Javanmard and A. Montanari, "On online control of false discovery rate," *arXiv preprint arXiv:1502.06197*, 2015.

[59] ——, "Online rules for control of false discovery rate and false discovery exceedance," *The Annals of Statistics*, vol. 46, no. 2, pp. 526–554, 2018.

[60] A. Ramdas, F. Yang, M. J. Wainwright, and M. I. Jordan, "Online control of the false discovery rate with decaying memory," in *Advances In Neural Information Processing Systems*, 2017, pp. 5650–5659.

[61] A. Ramdas, T. Zrnic, M. Wainwright, and M. Jordan, "SAFFRON: An adaptive algorithm for online control of the false discovery rate," in *International Conference on Machine Learning*, 2018, pp. 4286–4294.

[62] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE Symposium on Security and Privacy (S&P)*, 2017.

[63] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, "ML-leaks: Model and data independent membership inference attacks and defenses on machine learning models," in *Symposium on Network and Distributed System Security (NDSS)*, The Internet Society, 2019.

[64] C. Song and V. Shmatikov, "Overlearning reveals sensitive attributes," in *International Conference on Learning Representations (ICLR)*, 2020.

[65] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *ACM Conference on Computer and Communications Security (CCS)*, 2015, pp. 1322–1333.

[66] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song, "The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks," in *USENIX Security Symposium*, 2019.

[67] A. Salem, A. Bhattacharya, M. Backes, M. Fritz, and Y. Zhang, "Updates-leak: Data set inference and reconstruction attacks in online learning," in *USENIX Security Symposium*, S. Capkun and F. Roesner, Eds., USENIX Association, 2020, pp. 1291–1308.

[68] S. Zanella-Béguelin, L. Wutschitz, S. Tople, V. Rühle, A. Paverd, O. Ohrimenko, B. Köpf, and M. Brockschmidt, "Analyzing information leakage of updates to natural language models," in *ACM Conference on Computer and Communications Security (CCS)*, 2020.

[69] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, "Property inference attacks on fully connected neural networks using permutation invariant representations," in *ACM Conference on Computer and Communications Security (CCS)*, ser. CCS '18, Toronto, Canada: Association for Computing Machinery, 2018, pp. 619–633, ISBN: 9781450356930.

[70] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *IEEE Symposium on Security and Privacy (S&P)*, 2019.

[71] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning," *CoRR*, vol. abs/1912.04977, 2019. arXiv: 1912.04977.

[72] *Global AML and Financial Crime TechSprint*, https://www.fca.org.uk/events/techsprints/2019-global-aml-and-financial-crime-techsprint, [Online; accessed 25-Jan-2021], 2019.

[73] *Azure confidential computing, Microsoft Azure*, https://azure.microsoft.com/en-au/solutions/confidential-compute.

[74] *Microsoft SEAL (release 3.6)*, https://github.com/Microsoft/SEAL, Microsoft Research, Redmond, WA., Nov. 2020.

[75] P. Karnati, *Data-in-use protection on IBM cloud using Intel SGX*, https://www.ibm.com/cloud/blog/data-use-protection-ibm-cloud-using-intel-sgx, 2018.

[76] B. Knott, S. Venkataraman, A. Hannun, S. Sengupta, M. Ibrahim, and L. van der Maaten, "Crypten: Secure multi-party computation meets machine learning," in *Proceedings of the NeurIPS Workshop on Privacy-Preserving Machine Learning*, 2020.

[77] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, and G. Felici, "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers," *Int. J. Secur. Netw.*, vol. 10, no. 3, pp. 137–150, Sep. 2015.

[78] P. Mohassel and Y. Zhang, "SecureML: a system for scalable privacy-preserving machine learning," in *IEEE Symposium on Security and Privacy (S&P)*, 2017, pp. 19–38.

[79] V. Nikolaenko, S. Ioannidis, U. Weinsberg, M. Joye, N. Taft, and D. Boneh, "Privacy-preserving matrix factorization," in *ACM Conference on Computer and Communications Security (CCS)*, Berlin, Germany: Association for Computing Machinery, 2013, pp. 801–812, ISBN: 9781450324779.

[80] R. Gilad-Bachrach, K. Laine, K. Lauter, P. Rindal, and M. Rosulek, "Secure data exchange: A marketplace in the cloud," in *Proceedings of the 2019 ACM SIGSAC*

Conference on Cloud Computing Security Workshop, ser. CCSW'19, London, United Kingdom: Association for Computing Machinery, 2019, pp. 117–128, ISBN: 9781450368261.

[81] S. de Hoogh, B. Schoenmakers, P. Chen, and H. op den Akker, "Practical secure decision tree learning in a teletreatment application," in *Financial Cryptography and Data Security*, N. Christin and R. Safavi-Naini, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 179–194, ISBN: 978-3-662-45472-5.

[82] S. Laur, H. Lipmaa, and T. Mielikäinen, "Cryptographically private support vector machines," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06, Philadelphia, PA, USA: Association for Computing Machinery, 2006, pp. 618–624, ISBN: 1595933395.

[83] T. Graepel, K. Lauter, and M. Naehrig, "Ml confidential: Machine learning on encrypted data," in *Information Security and Cryptology – ICISC 2012*, T. Kwon, M.-K. Lee, and D. Kwon, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 1–21, ISBN: 978-3-642-37682-5.

[84] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa, "Oblivious multi-party machine learning on trusted processors," in *USENIX Security Symposium*, 2016.

[85] N. Hynes, R. Cheng, and D. Song, "Efficient deep learning on multi-source private data," *CoRR*, vol. abs/1807.06689, 2018.

[86] T. Hunt, C. Song, R. Shokri, V. Shmatikov, and E. Witchel, "Chiron: Privacy-preserving machine learning as a service," *CoRR*, vol. abs/1803.05961, 2018.

[87] F. Tramer and D. Boneh, "Slalom: Fast, verifiable and private execution of neural networks in trusted hardware," in *International Conference on Learning Representations (ICLR)*, 2019.

[88] R. Pass, E. Shi, and F. Tramèr, "Formal abstractions for attested execution secure processors," in *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, ser. Lecture Notes in Computer Science, vol. 10210, 2017, pp. 260–289.

[89] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology—CRYPTO*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 643–662, ISBN: 978-3-642-32009-5.

[90] S. Dov Gordon, F.-H. Liu, and E. Shi, "Constant-round mpc with fairness and guarantee of output delivery," in *Advances in Cryptology—CRYPTO*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 63–82, ISBN: 978-3-662-48000-7.

[91] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in *USENIX Security Symposium*, ser. SEC'11, San Francisco, CA, 2011, p. 35.

[92] T. Rabin and M. Ben-Or, "Verifiable secret sharing and multiparty protocols with honest majority," in *ACM Symposium on Theory of Computing (STOC)*, ser. STOC '89, Seattle, Washington, USA: Association for Computing Machinery, 1989, pp. 73–85, ISBN: 0897913078.

[93] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "GAZELLE: A low latency framework for secure neural network inference," in *USENIX Security Symposium*, 2018.

[94] C. Song, T. Ristenpart, and V. Shmatikov, "Machine learning models that remember too much," in *ACM Conference on Computer and Communications Security (CCS)*, ser. CCS '17, Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 587–601, ISBN: 9781450349468.

[95] F. Locatello, G. Abbati, T. Rainforth, S. Bauer, B. Schölkopf, and O. Bachem, "On the fairness of disentangled representations," in *Advances in Neural Information Processing Systems*, 2019, pp. 14 584–14 597.

[96] *Kaggle health dataset*, https://www.kaggle.com/c/hhp.

[97] R. Kohavi, "Scaling up the accuracy of Naive-Bayes classifiers: A decision-tree hybrid," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96, Portland, Oregon: AAAI Press, 1996, pp. 202–207.

[98] M. Lichman *et al.*, *UCI machine learning repository*, 2013.

[99] E. Creager, D. Madras, J.-H. Jacobsen, M. Weis, K. Swersky, T. Pitassi, and R. Zemel, "Flexibly fair representation learning by disentanglement," in *International Conference on Machine Learning (ICML)*, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97, 2019, pp. 1436–1445.

[100] *Yelp open dataset*, https://www.yelp.com/dataset.

[101] J. Leskovec, L. A. Adamic, and B. A. Huberman, "The dynamics of viral marketing," *ACM Transactions on the Web (TWEB)*, vol. 1, no. 1, 5–es, 2007.

[102] *Amazon product co-purchasing network metadata*, http://snap.stanford.edu/data/amazon-meta.html.

[103] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters," *Internet Mathematics*, vol. 6, no. 1, pp. 29–123, 2009. eprint: https://doi.org/10.1080/15427951.2009.10129177.

[104] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[105] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[106] K. Pearson, "Vii. note on regression and inheritance in the case of two parents," *proceedings of the royal society of London*, vol. 58, no. 347-352, pp. 240–242, 1895.

[107] H. Cramér, *Mathematical methods of statistics*. Princeton university press, 1999, vol. 43.

[108] D. J. Sheskin, *Handbook of parametric and nonparametric statistical procedures*. crc Press, 2020.

[109] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. M. Botvinick, S. Mohamed, and A. Lerchner, "Beta-vae: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations (ICLR)*, 2017.

[110] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork, "Learning fair representations," in *International Conference on Machine Learning (ICML)*, ser. ICML'13, Atlanta, GA, USA: JMLR.org, 2013.

[111] J. Hamm, "Preserving privacy of continuous high-dimensional data with minimax filters," in *Artificial Intelligence and Statistics Conference (AISTATS)*, 2015.

[112] H. Edwards and A. Storkey, "Censoring representations with an adversary," in *International Conference on Learning Representations (ICLR)*, Feb. 2016.

[113] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[114] G. Cormode, "Personal privacy vs population privacy: Learning to attack anonymization," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '11, San Diego, California, USA: Association for Computing Machinery, 2011, pp. 1253–1261, ISBN: 9781450308137.

[115] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM Conference on Computer and Communications Security (CCS)*, ACM, 2016, pp. 308–318.

[116] C. Song and V. Shmatikov, "Auditing data provenance in text-generation models," in *International Conference on Knowledge Discovery & Data Mining (KDD)*, ACM, 2019, pp. 196–206.

[117] X. He, J.-Y. Jia, M. Backes, N. Gong, and Y. Zhang, "Stealing links from graph neural networks," in *USENIX Security Symposium*, 2020.

[118] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning," in *ACM Conference on Computer and Communications Security (CCS)*, ser. CCS '17, Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 603–618, ISBN: 9781450349468.

[119] J. Buolamwini and T. Gebru, "Gender shades: Intersectional accuracy disparities in commercial gender classification," in *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, ser. FAT* '18, 2018, pp. 77–91.

[120] W. Zhang, S. Tople, and O. Ohrimenko, *Dataset-level attribute leakage in collaborative learning*, 2020. arXiv: 2006.07267 [cs.LG].

[121] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, "Fairness through awareness," in *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ser. ITCS '12, 2012, pp. 214–226.

[122] S. Song, Y. Wang, and K. Chaudhuri, "Pufferfish privacy mechanisms for correlated data," in *Proceedings of the 2017 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '17, 2017, pp. 1291–1306.

[123] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations (ICLR)*, Y. Bengio and Y. LeCun, Eds., 2014.

[124] X. He, A. Machanavajjhala, and B. Ding, "Blowfish privacy: Tuning privacy-utility trade-offs using policies," in *SIGMOD*, 2014, pp. 1447–1458.

[125] C. Louizos, K. Swersky, Y. Li, M. Welling, and R. S. Zemel, "The variational fair autoencoder," in *International Conference on Learning Representations (ICLR)*, Y. Bengio and Y. LeCun, Eds., 2016.

[126] A. Machanavajjhala and J. Gehrke, "On the efficiency of checking perfect privacy," in *Symposium on Principles of Database Systems (PODS)*, ser. PODS '06,

Chicago, IL, USA: Association for Computing Machinery, 2006, p. 163 172, ISBN: 1595933182.

[127] K. Atasu and T. Mittelholzer, "Linear-complexity data-parallel earth mover's distance approximations," in *Proceedings of the 36th International Conference on Machine Learning*, ser. ICML '19, 2019, pp. 364–373.